Faculdade de Engenharia da Universidade do Porto

Simulação de sistemas de produção industriais

Joaquim Graciano de Sousa Esteves

Dissertação realizada no âmbito do Mestrado Integrado em Engenharia Electrotécnica e de Computadores

Mestrado Integrado em Engenharia Electrotécnica e de Computadores

Major Automação

Orientador: Prof. Dr. José António Rodrigues Pereira de Faria
Junho 2009

Resumo

O trabalho desenvolvido teve por objectivo a criação de um modelo de simulação que ajude a tomada de decisões por parte dos responsáveis de produção de uma empresa industrial do sector metalomecânico.

Com a criação de um variado número de cenários que, embora idealizados, estão bastante próximos da realidade, pretende-se avaliar o sistema de produção de modo a permitir obter uma primeira definição do tamanho dos *buffers* dos produtos semi-acabados à saída do polimento e a uma primeira afectação das máquinas a uma, ou mais, referências, conforme os diferentes cenários de procura.

As ferramentas desenvolvidas permitem ao gestor efectuar análises de sensibilidade relativamente a factores como o tempo de *setup*, variação na procura, taxa de avarias e avaliar o seu impacto sobre o dimensionamento dos *buffers*.

Abstract

The goal of the work developed in this thesis was the creations of a simulation model that assists in the decision processes that the managers in charge of production have to do. The industry in this case is an electromechanical.

A number of idealized scenarios were created that are quite close to reality. The main idea was to evaluate the production system so that a first configuration for the buffer size for the semi-finished products (i.e., obtained after the polishing stage) and the assignment of one or more references to each machine, as part of a simulation scenario.

The tools developed allow the manager to perform sensitivity analysis on factors such as setup time, search variations, malfunction rates e evaluate their impact on the buffer settings.

Agradecimentos

A todos os que me ajudaram durante o meu percurso académico, em particular:

Ao meu orientador, Professor José Faria, pelos seus ensinamentos durante o percurso académico e em especial durante a realização da tese pela sua competência, conselhos e disponibilidade.

À minha família em especial à minha mãe que sempre me apoiou e incentivou.

Ao grupo de colegas e verdadeiros amigos que partilharam muitas noites de estudo na faculdade.

Índice

Lista de Figuras

Lista de Tabelas

Capitulo 1

1. Introdução

Este trabalho, foi desenvolvido no âmbito da disciplina de Dissertação do Mestrado Integrado em Engenharia Electrotécnica e de Computadores da Faculdade de Engenharia do Porto (FEUP), do ramo de automação industrial na área de especialização de Gestão Industrial e neste capítulo, pretende-se contextualizar o projecto, traçar os seus objectivos, mencionar a metodologia utilizada para a realização da dissertação bem como a estrutura do documento.

1.1 Contexto

Com a revolução tecnológica, o mundo passa por um processo de transformação muito rápido em que as empresas deixaram de competir no mercado interno para competir a nível mundial em que as empresas disputam os mercados globais. Isso levou á deslocalização de muitas empresas para países onde os custos de produção são mais baixos.

As empresas que permanecem nos países onde os custos de produção são mais elevados têm de fazer a diferenciação dos seus produtos pela qualidade e inovação tecnológica.

Com a introdução, em escala crescente, da informática, da robótica, das experimentações genéticas, das telecomunicações, a vida das pessoas tem mudado bastante. Essas mudanças também afectam a forma como o homem produz os bens necessários à sua sobrevivência.

Houve uma mudança de paradigma que passou da produção em massa para uma produção personalizada em que os bens são produzidos de maneira a satisfazer a maior diversidade possível de consumidores. O poder passou das grandes empresas que produzem em massa e em que o consumidor comprava o que eles produziam para o consumidor em que este compra o que quer tendo hoje uma grande gama de escolha de bens de consumo. Um exemplo disso é as empresas de telecomunicações tanto as produtoras de equipamento como as de serviços em que oferecem aos seus clientes as mais variadas gamas de equipamento e serviços.

As empresas tiveram de se adaptar rapidamente a esta nova filosofia de consumo e só as mais flexíveis a que conseguem ganhar quota de mercado neste novo mundo cada vez mais competitivo e global.

No sentido de adaptar as melhores práticas de mercado sem que isso afecte o normal funcionamento das empresas apareceu a simulação.

Com a simulação podemos simular as mudanças necessárias para que a empresa se possa preparar para os novos desafios testando os vários cenários propostos sem que isso interfira com o seu normal funcionamento e assim poupar tempo, dinheiro e ter uma adaptação mais consistente e preparada.

Neste documento é apresentado um estudo de simulação com base numa empresa do sector de metalomecânica que se dedica ao fabrico de torneiras, este estudo vai incidir no departamento de polimento e lixamento.

A empresa fabrica uma vasta gama de produtos com volumes de produção semanais muito variáveis.

Em anos recentes a empresa tem feito um esforço para evoluir no sentido da produção *lean,* nomeadamente através da evolução de fluxos *push para* fluxos *pull*. No entanto tem se deparado com dificuldades ao tentar criar os fluxos *pull* porque o sistema de produção está sujeito a múltiplos factores de instabilidade entre os quais taxas de avarias de equipamento, tempos de *setup* elevados e com grande variabilidade e grande flutuação na procura.

Nestas condições, e em consequência destes factores algumas tentativas para introdução de fluxos puxados fracassaram pois as instabilidades acima mencionadas obrigam a frequentes desvios face aos fluxos produtivos planeados.

Por outro lado, no caso dos produtos cujos valores de produção estavam relativamente estabilizados através do nivelamento da procura foi possível evoluir com sucesso para um fluxo de produção puxada.

Assim uma questão fundamental que se coloca aos responsáveis da produção da empresa é saber que referências devem funcionar em *pull* e que referências devem funcionar em *push*.

Está relacionado com esta decisão a afectação das máquinas, isto é, se uma das máquinas funciona apenas com uma referência ou se há a possibilidade da mesma máquina variar as referências a produzir.

Estas duas alternativas serão avaliadas para diferentes valores da procura. No caso da procura ser estável a máquina produzirá para uma referência já se a procura for muito variável cada máquina terá de produzir várias referências.

Pretende-se a minimização de *stocks* e sua dependência dos tempos de *setup* para vários cenários de procura.

1.2 Objectivos

O objectivo deste trabalho é testar vários cenários em processos de produção utilizando uma filosofia l*ean* com base numa ferramenta de simulação, o AnyLogic.

Pretende-se desenvolver modelos de simulação que permitam aos responsáveis de produção encontrar as soluções mais eficientes para o sistema de produção permitindo-lhes decidir sobre

aspectos como a afectação das máquinas a uma ou mais referencias e o dimensionamento dos *buffers*.

1.3 Metodologia

Para alcançar os objectivos propostos, o trabalho desenvolvido foi dividido basicamente em duas partes. Inicialmente foi um estudo teórico sobre o tema, aprofundando os conhecimentos de produção *Lean*, simulação e estudando a ferramenta de simulação utilizada, *AnyLogic*. Ainda nesta fase foram desenvolvidos um conjunto de aplicações de simulação com o objectivo de dominar a ferramenta. Na segunda parte foram desenvolvidos modelos de simulação para diferentes cenários de procura e tempos de *setup*.

1.4 Organização da dissertação

A presente tese encontra-se estruturada em 6 capítulos, cujo conteúdo é sumariamente descrito em seguida:

No primeiro capítulo é feita a introdução ao trabalho, quais os objectivos, a metodologia seguida na sua realização e apresentação da estrutura do documento.

No segundo capítulo é feito uma descrição do estado da arte do tema em questão, simulação, modelos de simulação e principais áreas de actividade onde a simulação é mais utilizada.

No terceiro capítulo é feito uma descrição da ferramenta utilizada para a realização do trabalho, o AnyLogic.

No quarto capítulo é feita uma descrição global do sistema de produção e onde se faz um primeiro modelo de simulação. Aqui, além da descrição do sistema físico idealizado, é apresentado a respectiva modelação, tanto na componente estrutura como na componente dinâmica.

São descritos os vários *active object* e classes java criados.

No capítulo quinto é desenvolvido um novo modelo de simulação para simular cenários de procura e a sua influência no dimensionamento dos *buffers* bem como na taxa de utilização das máquinas. Verificar qual a melhor opção relativamente á afectação das máquinas, se dedicada a apenas á produção de uma referencia ou produzir várias referências.

No sexto e último capítulo é feita a uma avaliação geral do trabalho.

Capitulo 2

2. Simulação

Neste capitulo é feita uma discussão sobre simulação, quais as suas vantagens e desvantagens, como se constrói um modelo e quais os objectivos da simulação e sua implementação.

2.1 Introdução

Os métodos de simulação em computadores tiveram seu início no começo da década de 60, e sua origem coincidiu com a introdução dos computadores no mercado mundial. Por ser uma técnica que necessita de uma grande quantidade de cálculos matemáticos, sua viabilização sem um computador seria impossível. Inicialmente, a simulação foi usada nos EUA, para planeamento de operações militares. Nessas aplicações, os principais usos eram o planeamento de distribuição de suprimentos às frentes de batalha (Nº de navios/veículos) e alocação de recursos escassos (artilharia). O sucesso alcançado foi enorme.

Percorrendo a internet podemos ver o que significa simulação na wikipedia, “Em computação, simulação consiste em empregar técnicas matemáticas em computadores com o propósito de imitar um processo ou operação do mundo real. Desta forma, para ser realizada uma simulação, é necessário construir um modelo computacional que corresponda à situação real que se deseja simular”.

Quando se recorre á simulação utilizando ferramentas informáticas consiste em empregar técnicas matemáticas em computadores com o propósito de imitar um processo ou operação do mundo real. Desta forma, para ser realizada uma simulação, é necessário construir um modelo computacional que corresponda à situação real que se deseja simular.

Assim, a simulação computacional de sistemas, ou apenas simulação é o estudo do comportamento de sistemas reais através do exercício de modelos.

Neste trabalho a ferramenta informática que foi utilizada para simular e criar o modelo foi o programa informático *AnyLogic*.

A simulação é um processo de projectar um modelo computacional de um sistema real e conduzir experiencias com este modelo com o propósito de entender seu comportamento e/ou avaliar estratégias para sua operação, (Pegden, 1995). Desta maneira, podemos entender a simulação como um processo amplo que engloba não apenas a construção do modelo, mas todo o método experimental.

Para entendermos melhor o que é simulação, precisamos também conhecer as definições de sistemas e modelos. Um sistema é um conjunto de elementos distintos, que exercem entre si uma interacção ou interdependência. Por natureza, os sistemas são limitados, ou seja, deve-se definir limites ou fronteiras. Portanto, podemos definir sistemas dentro de outros sistemas, e assim por diante. Um modelo, é uma representação de um *sistema* real, na qual somente os aspectos relevantes para uma determinada análise deste sistema são considerados.

Na simulação computacional, estamos particularmente interessados nos modelos simbólicos, que usamos para representar um sistema real em um computador. O principal ponto que temos que considerar aqui, é que um modelo é criado para representar alguma coisa do mundo real.

Com o desenvolvimento das novas tecnologias a simulação tem um relevo cada vez maior, já que existem ferramentas cada vez mais poderosas capazes de modelar os mais complicados processos do mundo real sem que para isso seja necessário interferir com o normal funcionamento do mundo real que pretendemos simular.

Uma empresa quando pretende fazer alterações nos processos de produção a melhor maneira de prever como irá funcionar será modelar os processos em causa e simular as várias situações que podem ocorrer com a transformação dos processos em causa sem que para isso tenho de correr os riscos de uma implementação prematura dos processos no mundo real com todas as despesas e transtornos que poderia causar.

Um dos factores de competividade das empresas é a sua capacidade de absorverem as melhores práticas existentes sem que isso coloque em causa a sua capacidade económica por uma transformação de processos que seja mal planeada, assim sempre que possível devem fazer uma simulação dos novos processos a implementar e avaliar as vantagens sem colocar em risco os processos existentes.

A Simulação torna possível o estudo, a análise e a avaliação de diversas situações que não poderiam ser conhecidas de outro modo. Num mundo cada vez mais competitivo, a simulação tornou-se numa metodologia de resolução de problemas indispensável para a industria, serviços, transportes, logística, etc.

Com a simulação computacional podemos fazer os mais variados cenários para o modelo em causa sem que para isso a empresa tenho de correr o risco de implementar esses cenários no mundo real com todas as despesas que essa implementação acarreta.

Quando implementamos uma simulação devemos ter em atenção os seguintes passos.

- Uma correcta formulação dos problemas
- Definição de objectivos e abordagem geral
- Construção do modelo e recolha de dados
- Codificação do modelo
- Verificação do modelo codificado
- Validação
- Projectar a simulação
- Executar a simulação e análise dos dados
- Documentar o modelo e fazer o relatório
- Implementar os resultados.

2.2 Modelos de simulação

Modelo: É uma representação simplificada de um sistema (ou processo ou teoria) com o propósito de melhorar a nossa capacidade de entender, prever e possivelmente controlar o sistema.

Modelagem mental é uma actividade humana básica que simplifica os processos de planeamento e tomada de decisão. Os modelos são úteis pois a construção física do sistema real pode ser impossível, muito cara, pode consumir muito tempo, ou ainda, a realização de experiencias com o sistema pode ser perigosa ou destrutiva.

Como exemplos de sistemas difíceis de serem tratados directamente:

- Reactores nucleares;
- Sistema solar;
- Sistemas económicos;
- Sistemas que envolvam riscos humanos.

A representação de um modelo pode ter diferentes formas, ele pode ser mental, físico ou simbólico.

No modelo mental é o processo pessoal de tomada de decisão em que á uma visão pessoal de um objecto, evento que nos leva a criar um modelo mental.

No modelo simbólico é a explicação de fenómenos físicos e químicos por formulas matemáticas como por exemplo H_2 + O representa a fórmula química da água, F = ma representa uma força que é igual á massa x aceleração, a representação de um circuito eléctrico, etc.

Modelos físicos são construídos com componentes concretos (tangíveis). São representações de sistemas físicos como sistemas eléctricos, térmicos, hidráulicos, etc. São descritos por variáveis como voltagens, correntes, potência, temperaturas, comprimentos, pressão, fluxo, força, velocidade, etc.

Os modelos físicos podem ser classificados em estáticos em que as variáveis não variam ao longo do tempo e dinâmicos em que as variáveis podem se alterar no decorrer do tempo.

A figura ilustra os vários modelos físicos até chegar ao modelo de simulação.

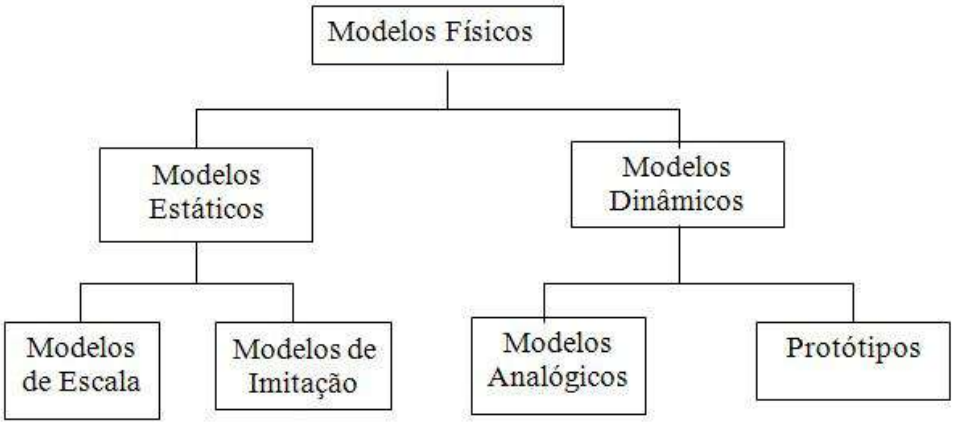

Figura 1 - Modelos físicos

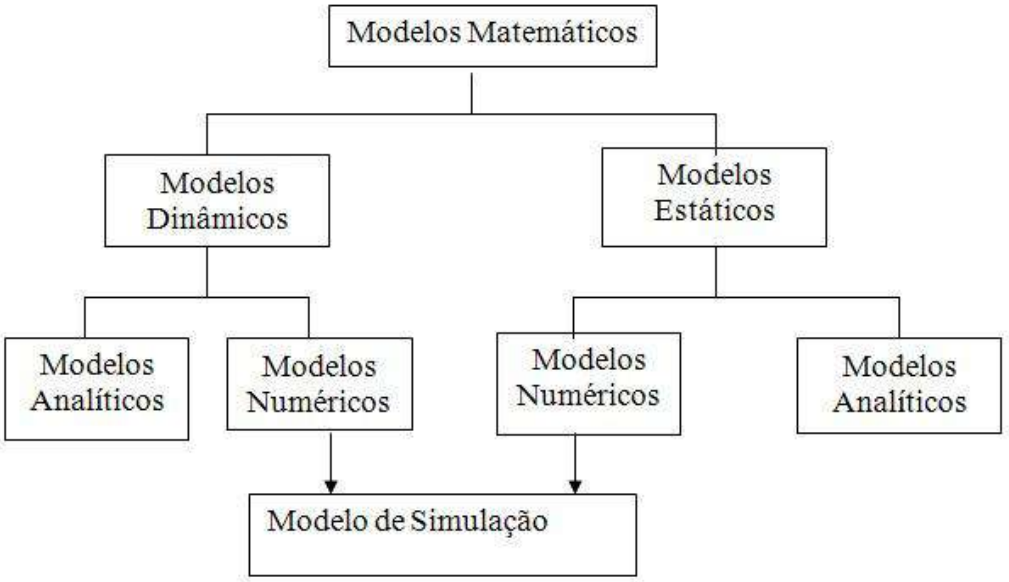

Figura 2- Modelos matemáticos

2.3 Áreas onde a simulação é mais utilizada.

Devido à sua versatilidade, flexibilidade e poder, a simulação pode ser aplicada em qualquer estudo ou pesquisa na utilidade e uso de técnicas de pesquisa operacional. Quase todos os tipos de

sistemas foram (ou podem ser) simulados, e a larga faixa de aplicações quase extrapola a classificação.

Um bom exemplo de simulação é aquele usado na indústria aeronáutica, onde a aerodinâmica dos aviões em projecto é testada em túneis de vento através de pequenas maquetas que apresentam o mesmo formato do avião, ou seja, é o "modelo" do avião real. Esta técnica é aplicada, pois seria completamente inviável construir todo o avião e tentar fazê-lo voar com pilotos de prova. A perda de vidas e investimentos seria enorme e certamente nossos aviões não seriam como hoje os conhecemos se não fosse usada a simulação.

A evolução vertiginosa da informática nos últimos anos tornou o computador um importante aliado da simulação. A simulação por computador é usada nas mais diversas áreas tendo um papel muito importante na simulação de voo.

Um simulador de voo é um sistema de aparelhos ou um software que pretende recriar o voo de uma aeronave da maneira mais realista possível.

Os simuladores profissionais são aparelhos complexos com sistemas de actuadores hidráulicos ou eléctricos para recriar os movimentos de uma aeronave durante o seu voo. Servem como um instrumento de ensino, treino, e em alguns casos ajudam as autoridades a investigar as causas de acidentes aéreos.

A sua função também é de redução de custos com manutenção, desgastes e reciclagem de pilotos em situações normais e adversas como treino de condições críticas. Seu uso possibilita economia de até 1/8 do custo da hora de uma aeronave. O treino de Procedimentos de Cabine e Navegação Aérea Simuladores Parciais, que podem incluir sistema de movimento, controle de comandos (*Control Loading*), e Sistemas Visuais. FFS - *Full Flight* Simulator, que são simuladores completos, que simulam o voo bem como reproduz o mundo virtual da área a ser treinada com imagens de satélite e objectos 3D. Sistema de movimento com alto grau de realismo.

Outro exemplo de utilização da simulação é as previsões meteorológicas em que através de modelos se consegue prever de uma força bastante fiável o tempo que irá ocorrer nos próximos dias.

Sem recursos a simulação não seria possível prever como serão as condições atmosféricas nos próximos dias já que é um fenómeno que ainda nem aconteceu.

Parece um exemplo banal mas tem grande importância em vários sectores de actividade tanto a nível económico como social.

No dimensionamento de serviços de atendimento ao público é outra área de grande interesse na simulação como é o caso de dimensionamento de uma central de atendimento em que um software de simulação cria uma central de atendimento virtual e nela se pode verificar a eficiência do negócio e eficácia do atendimento ser ter de interferir com o normal funcionamento da central de atendimento real. Pode alterar as várias variáveis que têm influência no desempenho da actividade tais como necessidade de mais equipamento e pessoal, se tem congestionamentos de tráfego ou se um aumento de 30% de tráfego provocaria congestionamento.

Nos Sistemas Computacionais como os componentes de hardware, software, redes de computadores, estruturas e gestão de banco de dados, processamento da informação, confiabilidade de hardware e software.

Na área dos negócios, análise de mercado, política de preços, estratégias de marketing, estudos de aquisição de empresas, análise de fluxo de caixa, previsão, alternativas de transporte, planeamento de força de trabalho.

Nas áreas de actuação dos governos, armamentos e tácticas militares, previsão de crescimento populacional, uso do solo, sistemas de saúde, sistemas contra incêndio, polícia, justiça criminal, projectos de estradas, controle de tráfego, serviços de saneamento.

2.3.1 Simulação na produção.

Métodos tradicionais de projecto e análise têm simplesmente se mostrado inadequados para o estudo de complexas interacções que ocorrem nos sistemas de produção. As organizações estão a recorrer cada vez mais à simulação, como um veículo de análises dinâmicas antes da implementação de projectos.

A simulação prevê o comportamento de sistemas complexos calculando o movimento e interacção dos componentes do sistema. Pela avaliação do fluxo de peças através das máquinas e postos de trabalho, examinando os conflitos de procura por recursos limitados, podemos avaliar *Layouts*, selecção de equipamentos, e procedimentos de operação. A simulação fornece a capacidade de experimentar no modelo preferivelmente ao sistema real.

Pegden especifica que as ferramentas de simulação podem ser empregadas tanto nas fases de planeamento, quanto nas fases de projecto ou de operação de sistemas de produção, agindo sobre três áreas, a saber:

- *Hard-system:* concerne ao número e aos tipos dos equipamentos, a configuração do *layout* e as peças a serem processadas pelo sistema;
- *Soft-system*: refere-se ao planeamento, sequência das peças, *scheduling*, ferramentas, trabalhadores;
- *Controlo em tempo real:* controle de fluxo envolvendo contingências como falha de ferramenta, máquinas ou paradas, etc.

A utilização da simulação no futuro se dará em três modos distintos:

Como ferramenta de projecto e análise de *layouts*, de compra de máquinas, de avaliação de políticas alternativas, etc. Actualmente, estas são as funções da simulação mais frequentemente utilizadas.

Como ferramenta de *scheduling*, particularmente em se tratando de sistemas automatizados. Este uso permite ao tomador de decisão explorar, planear e avaliar mudanças no *scheduling* a fim de obter a optimização do mesmo.

Como parte de sistemas em tempo real, ou de controlo *on-line*. Tal sistema deverá ser periodicamente activado, ler as condições correntes, definindo o *scheduling*, com ou sem a intervenção humana.

Quando a simulação é usada como ferramenta de projecto, o estudo é tipicamente motivado por questões como:

- Qual será a capacidade de produção do projecto?
- Vai de encontro aos objectivos propostos?
- Onde estão os gargalos?
- O que pode ser mudado para aumentar a capacidade?
- Qual é a melhor dentre as várias alternativas de projecto?
- Como a performance do sistema muda como função do número e tipo das máquinas, número de operadores, tipos de automação?
- Qual é a confiabilidade do sistema?
- As quebras de máquinas afectarão a capacidade?

2.4 Vantagens e desvantagens da simulação

Devido ao fato de ser facilmente compreendido, um modelo de simulação é frequentemente mais simples de ser justificado do que alguns modelos analíticos. Além disto, um modelo de simulação costuma ter mais credibilidade, uma vez que pode ser comparado com o sistema real, ou porque requer pouca simplificação, capturando as características reais do sistema.

Todos os modelos de simulação são chamados de modelos de entrada e saída, isto é, eles produzem uma saída dadas as condições de entrada. Eles não podem gerar por si mesmo uma solução óptima como é o caso dos sistemas analíticos. Os modelos de simulação servem apenas como ferramenta de análise do comportamento do sistema sob determinadas condições.

Quando se decide pela simulação de um sistema do mundo real é porque esta tem grandes vantagens destacando as seguintes:

- A simulação proporciona uma melhor compreensão do sistema real;
- Poupança de tempo no processo de simulação, o que poderia demorar meses no sistema real pode ser transformado em minutos de simulação;
- Não interfere na operação do sistema real;
- É mais interessante que modelos matemáticos;
- Pode ser utilizado para treino de pessoal como é o caso dos simuladores de voo;
- As análises são mais realistas que a modelagem matemática;
- Pode ser utilizado em modelos transitórios, não apenas em regimes permanentes;
- Disponibilidade de variado software comercial para simulação;
- Análise de cenários alternativos;

- Simplificação das hipóteses;
- Suposições sobre os comportamentos;
- Prever o impacto de uma alteração no sistema real;
- Estudo de alternativas para modificar o sistema real;
- Estudar sistemas que não existem;
- Estudo de situações improváveis;
- Análise de alterações no funcionamento e na estrutura do sistema;
- Analisar situação e alternativas que não existem;
- Visualização do funcionamento de cada alternativa estudada;
- Avaliar uma proposta quando os dados de entrada são insuficientes;
- Comprimir ou expandir o tempo durante a simulação que de outra maneira não seria possível e assim permite observar fenómenos complexos com mais detalho;
- Economia de recursos financeiros, materiais e pessoal;
- Permite a análise de quais variáveis são significativas para o desempenho do sistema e como estas variáveis se interagem;
- Identificação de gargalos na cadeia de produção;

Deve-se ressalvar algumas restrições ou dificuldades na implantação de um modelo de simulação. As principais são:

- Necessidade de treino, uma vez que a qualidade da análise depende da qualidade do modelo, e portanto da habilidade do analista;
- Incerteza da solução óptima;
- Por vezes á dificuldades de validação dos modelos;
- Dificuldades na elaboração dos modelos que podem demorar bastante tempo;
- Análise estatística dos resultados;
- Tomada de decisão baseada numa simulação deficiente e não devidamente validada;
- A construção de modelos é uma arte que requer treino especializado. A qualidade da análise depende directamente da qualidade do modelo e da qualidade do modelador;
- Os resultados da simulação são algumas vezes difíceis de interpretar;
- O modelo de um sistema complexo pode ter um custo elevado e levar vários meses para ser desenvolvido, especialmente nos casos em que os dados são de difícil obtenção e não coerentes.

Capitulo 3

3. Apresentação do AnyLogic

3.1 Introdução

Neste capítulo é apresentada a ferramenta de simulação utilizada no trabalho, o AnyLogic.

O AnyLogic é uma ferramenta de simulação que funciona nos vários sistemas operativos (Windows, Linux, Mac) que permite a modelagem de sistemas através de diferentes métodos como: Dinâmica de sistemas, simulação orientada a eventos discretos e Modelação Baseada em agentes. Estes métodos podem ser combinados entre si e utilizados em simultâneo.

Para que o utilizador se possa ambientar com os métodos atrás referidos o AnyLogic tem tutoriais para cada um destes métodos.

Com o AnyLogic é possível simular as mais distintas situações, desde serviços a processos industriais em que se destacam os seguintes:

- Sistemas de Produção;
- Logística e transportes;
- Tráfico rodoviário;
- Processos de negócio;
- Saúde;
- Centros de atendimento;
- Área militar e Aeroespacial;
- Economia e gestão;
- Sistemas e emergência;
- Indústria automóvel;
- Comportamentos sociais.

No sítio de internet do AnyLogic podemos ver exemplos de aplicação e demos de modelos de simulação do AnyLogic. Estes demos comprovam a capacidade de simulação desta ferramenta nas mais variadas situação e ramos de actividade.

O AnyLogic tem também uma vertente estatística onde se podem analisar os dados recolhidos no decorrer da simulação com visualização gráfica dinâmica.

A linguagem de programação utilizada no AnyLogic é o Java com a componente de modelagem UML (*Unified Modeling Language*). Sempre que necessário podemos construir código para adicionar ao AnyLogic o que o torna muito flexível.

O Java é uma linguagem de programação orientada a objecto (POO).

Active Objects são os principais objectos utilizados no AnyLogic para a modelação e simulação de sistemas, são constituídos por parâmetros, funções, variáveis, eventos ou mesmo outros active objects.

3.2 Ambiente de modelação e simulação.

Neste tópico vamos descrever todos os blocos que compõem o ambiente de modelação onde vamos desenvolver os vários componentes a ser utilizados no trabalho proposto.

Na linha horizontal na parte superior tem os vários menus do AnyLogic como o File para abrir trabalhos anteriores e salvar os trabalhos em aberto. *Edit* para ajuda como apagar, corrigir e anular erros, fazer cópia de elementos no editor gráfico. *View* onde podemos seleccionar o que queremos que apareça na visualização do ecrã tais como Project *view*, *properties view*, *console view*. Os outros menus são de menor relevância e não serão mencionados.

Na figura 1podemos ver a forma como é apresentado o *AnyLogic* e os vários componentes que passamos a descrever.

- *Project* – esta janela fica situado na parte esquerda a figura, nele podemos ver os vários projectos abertos, a criação de novos projectos e a criação de novos active objects e classes Java dentro dos projectos abertos.
- Editor gráfico – Janela maior ao cento da figura onde é feita a modelação de cada active object, onde se faz a programação gráfica e a ligação entre os vários elementos do active object e outros active object do projecto.
- *Properties* – Janela abaixo do editor gráfico onde podemos ver e modificar as propriedades dos vários componentes criados no editor gráfico.
- *Console* - logo abaixo do *properties*, é onde aparecem os erros quando se corre o programa.
- *Palette* – fica no lado direito da figura, é onde estão as bibliotecas do AnyLogic e é constituído por:
    - *Model* - onde se encontram variáveis, parâmetros, eventos, funções, gráficos de estado e suas transições, etc.

- *Action* – não foi utilizado neste trabalho, contém funções para representar diagramas.
- *Analysis* – biblioteca para análise de dados, que podem ser representados por gráficos, histogramas, etc.
- *Presentation* – biblioteca para apresentações, contém figuras geométricas, botões, pode adicionar imagem, mensagem de texto, linhas para fazer desenhos, etc.
- *Connectivity* – Serve para fazer a ligação do projecto a uma base de dados
- *Enterprise Library* – biblioteca com componentes para simulação de sistemas orientados a eventos discretos.

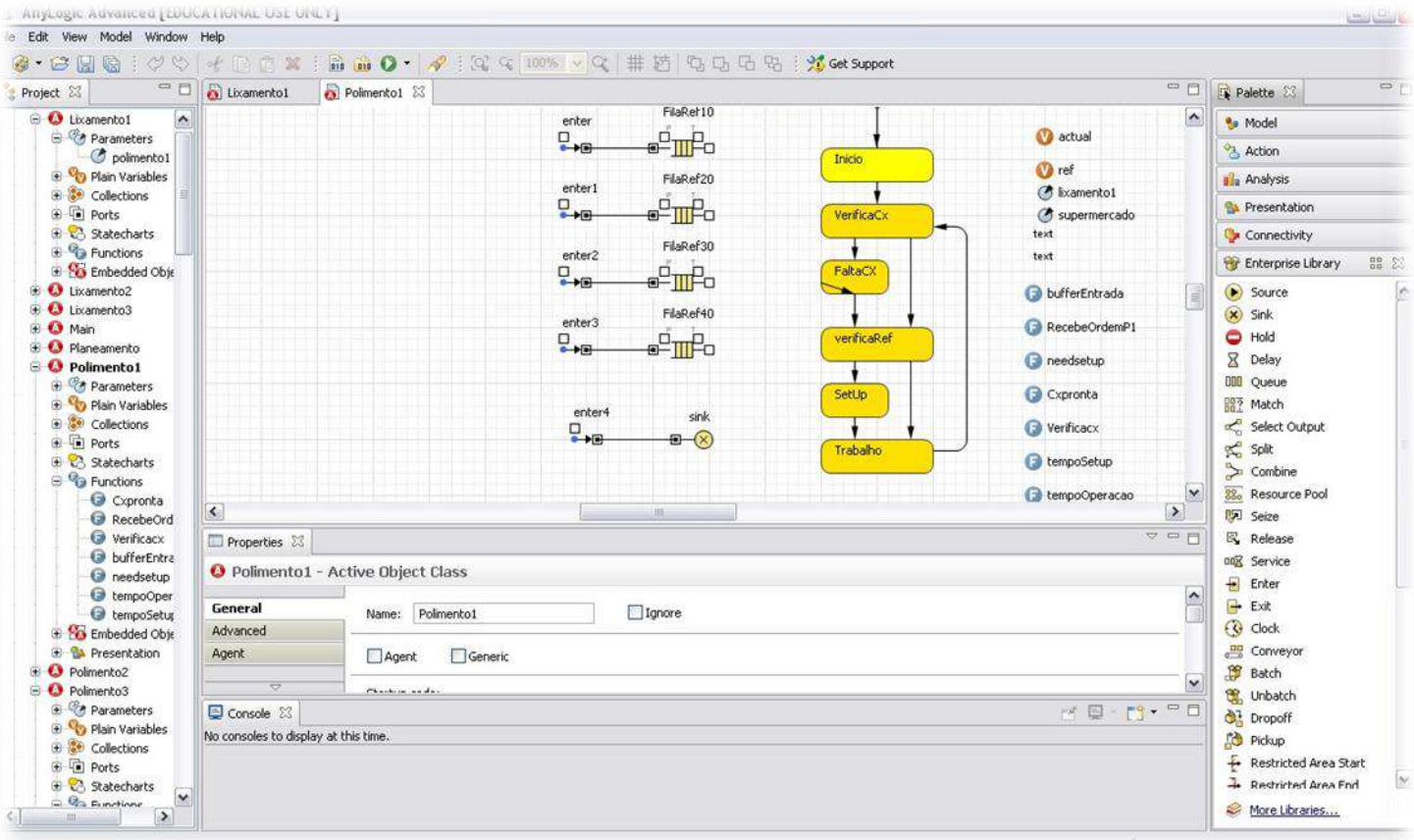

Figura 3- Área de trabalho do *AnyLogic*

3.3 Active object e classe Java.

O *active object* é onde se estrutura todo o modelo, podemos modelar as mais variadas situações do mundo real. Dentro do mesmo projecto podemos ter vários *active object* simulando as várias situações de uma empresa como o movimento de pessoas e fluxos, interacção entre os vários departamentos, etc.

O *active object* é modelado no editor gráfico descrito anteriormente, pode conter componentes das várias bibliotecas existentes na *Palette*.

A criação *do active object* é feita clicando com botão direito do rato no projecto a implementar na janela de *Project* onde abre uma janela para criação do novo *active object* classe. O procedimento é igual para a criação de uma classe java.

A figura 2 seguinte ilustra um active objecto utilizado no desenvolvimento do trabalho.
Em cima mostra os componentes da *Entreprise Library* como o *enter* , *queue e sink* .
Os diagramas de estado e as funções pertencem à biblioteca *model* .

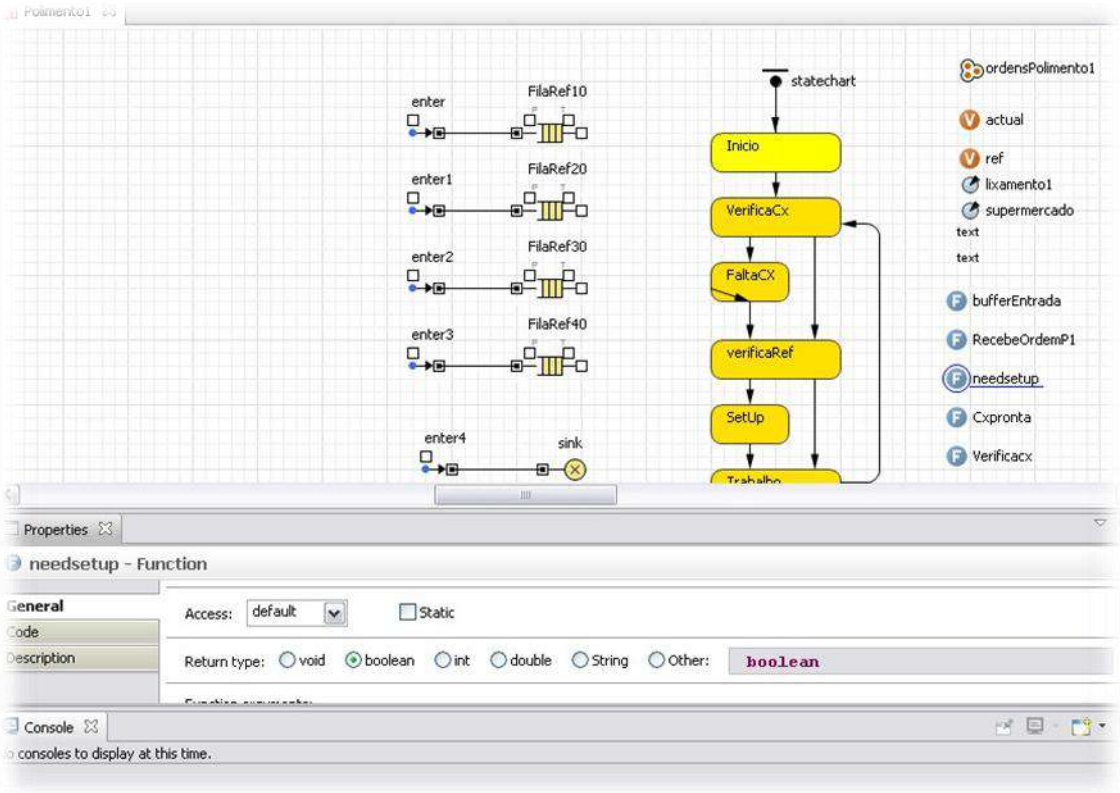

Figura 4- Active objecto criado no trabalho

A função needsetup ilustrada na figura 3 é do tipo booleano e faz a comparação das 2 ordens de produção, retornando a informação *true* para o caso das ordens serem de referências diferentes.

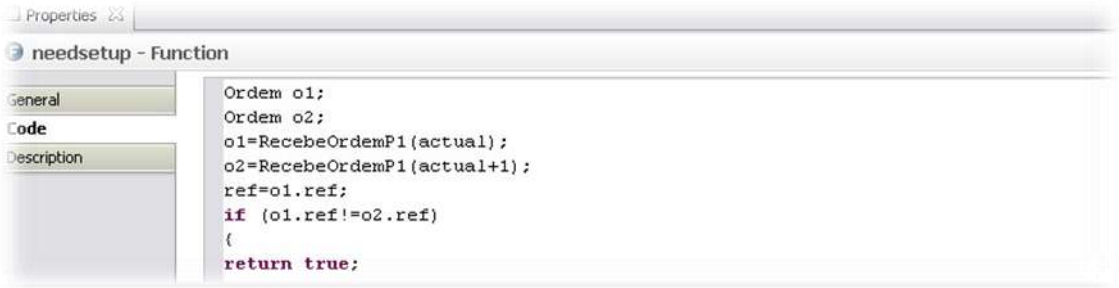

Figura 5- Função needsetup

A criação de uma classe java ilustrada na figura 4 é muito parecido com a criação do active object mas visualmente diferente. Aparece uma janela para a criação da classe java para que lhe seja atribuído o nome e a que super classe pertence, o passo seguinte é inserir os campos da classe, neste caso os campos são ref do tipo *string*, quantidade do tipo inteiro e ordem do tipo inteiro. Os métodos construtor e *toString* podem ser gerados automaticamente.

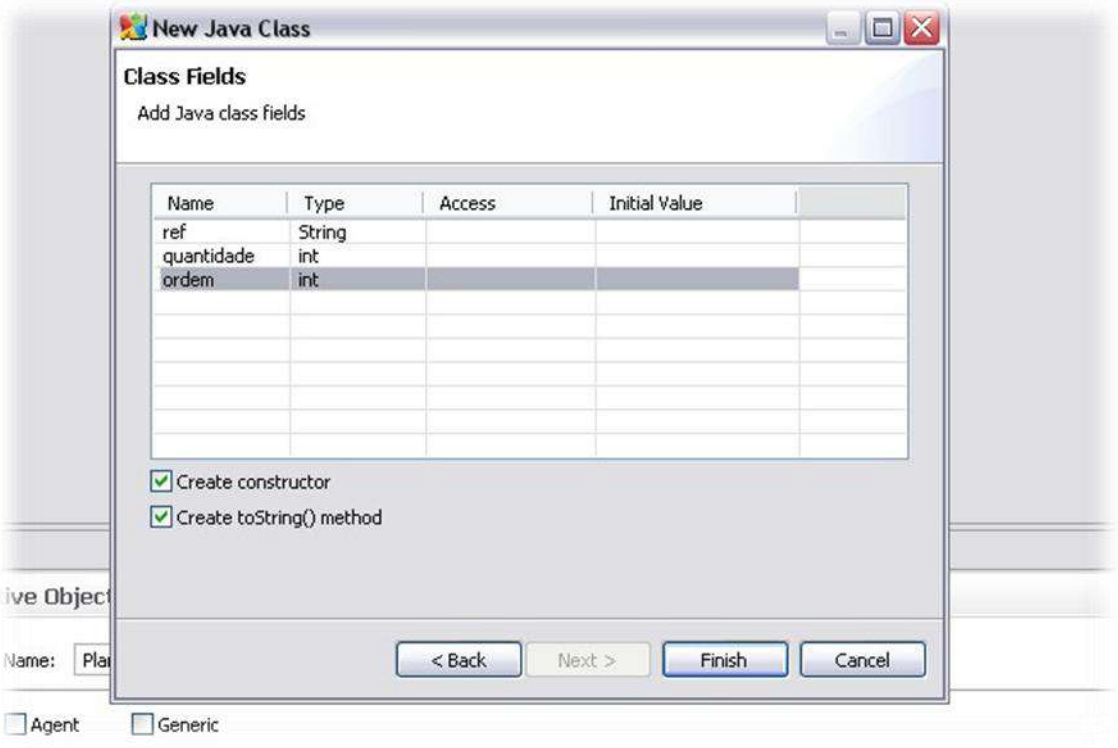

Figura 6- Criação de uma classe java

Quando se finaliza a criação da classe java aparece a figura 5 em que o corpo da classe aparece já construído e que poderemos alterar colocando ou retirando código se tal for conveniente.

MyClass.java

```
public class MyClass extends sistema_de_producaov3.Box implements java.io.Serializable {

    String ref;

    int quantidade;

    int ordem;


   /**
    * Default constructor
    */
   public MyClass(){
   }

   /**
    * Constructor initializing the fields
    */
   public MyClass(String ref, int quantidade, int ordem){
       this.ref = ref;
       this.quantidade = quantidade;
       this.ordem = ordem;
   }

   @Override
   public String toString() {
       return
           "ref = " + ref +" " +
           "quantidade = " + quantidade +" " +
           "ordem = " + ordem +" ";
   }
```

Figura 7- Corpo da classe java e respectivo código

3.4 Entreprise Library

A *Entreprise Library* foi a biblioteca que mais contribuiu para o desenvolvimento do trabalho e por esse facto terá uma atenção especial na explicação dos seus componentes, especialmente os mais utilizados no trabalho.

Orientada a eventos discretos, esta biblioteca é poderosíssima pois é capaz de simular os sistemas mais complexos através de fluxogramas que pela sua flexibilidade são uma importante ferramenta de simulação. Estes componentes podem representar o que quisermos desde movimentos de pessoas e fluxos de materiais a coisas mais subjectivas, como o comportamento das pessoas que se pode alterar por determinados parâmetros.

Com os diferentes componentes é possível representar movimentos entes os componentes da biblioteca seja através de portas de entrada e saída que ligam os vários componentes, seja através de outras operações como *take* e o remove que colocam e retiram entidades directamente dos componentes que aceitam estas funções.

A criação dos modelos é de fácil concepção já que o AnyLogic utiliza a técnica do *drag-and-drop*, em que se arrastam os componentes da *Palette* e são largados no editor gráfico para depois se fazerem as ligações entre eles. A ligação entre os componentes é feita através de duplo clique na porta de saída e depois outro duplo clique onde se quer ligar o componente.

Por defeito é a entidade Entity que percorre os vários componentes do modelo mas que pode ser alterada através da criação de classes java e de definição de nova entidade. Estas novas classes java são sempre subclasses da classe Entity

A cada componente estão associados eventos, *onEnter* e *onExit*, que ocorrem quando a entidade entra ou sai do referido componente e que podem chamar funções ou simulação gráfica associada à entrada ou saída da entidade.

3.4.1 Componentes da Enterprise Library

Nesta secção daremos relevância aos componentes mais utilizados no trabalho com figuras dos vários componentes e uma breve descrição do seu funcionamento.

Na figura 6 podemos ver todos os componentes pertencentes à Enterprise Library.

Figura 8- Objectos disponíveis na Entreprise Library

Source e Sink – A Source normalmente está no inicio do fluxograma e tem como função gerar entidades que vão percorrer o fluxograma. A *Sink* normalmente está no final do fluxograma e serve para eliminar entidades.

Na figura 7 podemos ver a maneira com são ligados a outros componentes, ambos Têm apenas uma porta de ligação, no caso da *source* uma saída e do sink uma entrada.

A *source* pode gerar entidades de qualquer subclasse da classe *entity*, essas entidades podem ser geradas de duas maneiras distintas, em modo manual através do comando *inject*() e no modo automático. No modo automático pode gerar as entidades em intervalos de tempo específicos ou através de uma lei de distribuição em que gera esses intervalos de tempo de maneira aleatória.

Podemos especificar quantas entidades são geradas de cada vez e associar uma animação no evento de saída da *source*, *onExit*, à entidade criada.

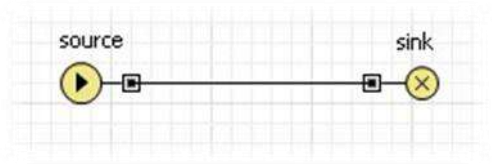

Figura 9- Componentes *source* e *sink*

Queue – este componente funciona como buffer em que guarda as entidades que nele entram de modo ordenados seguindo a ordem FIFO em que a primeira que entra é a primeira a sair ou em modo de prioridades em que cada entidade é avaliada em relação ás restantes e colocada na sua ordem de saída, este método no é menos utilizado que o primeiro.

A entrada na *queue* é feita de modo a seguir a ordem do fluxograma, a saída de entidades pode ser de duas maneiras, seguir a ordem do fluxograma ou através do método *remove*() em que a entidade é removida do *queue* e colocada onde se pretende.

A capacidade da *queu* pode ser definida tal como a sua animação, quando a atingida a sua capacidade máxima é gerado um erro.

A *queue* tem quatro portas, uma de entrada e três de saída, das três de saída uma é normal que liga ao objecto seguinte do fluxograma e as outras duas são portas especiais.

Porta *outTimeout* (representada por T na figura 8) retira a entidade do buffer depois de um tempo definido em que poderia lá permanecer.

Porta *outPreempted* (representada por P na figura 8) quando activa, empurra a ultima entidade a entrar para fora do buffer se este está cheio, sem seguir o fluxograma.

A *queue* tem três eventos associados ás suas portas de entrada e saída que são o *On enter, On exit* e *On at exit,* este ultimo é activo quando a entidade chega à saída do *buffer* mas ainda não saiu.

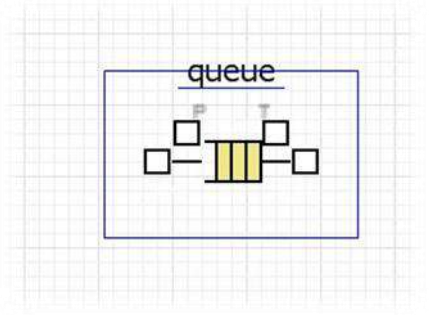

Figura 10- *Queue*

Enter e exit – Estes objectos são utilizados para transferir entidades entre diferentes partes do fluxograma, diferentes fluxogramas ou mesmo entre diferentes active object.

A entidade é colocada no *enter* através da função do *anyLogic take*(). Normalmente o *enter* está ligado ao objecto *queue* onde fica armazenada a entidade.

O objecto *enter* tem o evento associado à porta de entrada, *On enter*, e o objecto *exit* tem o evento associado à sua porta de saída, *On exit*. Estes eventos podem servir para simular a entrada e saída de entidade no decorrer da simulação.

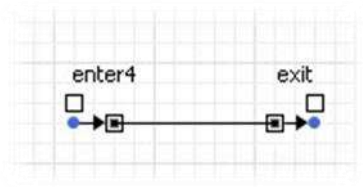

Figura 11- Componentes *enter* e *exit*

Delay e hold – O objecto *delay* tem como função reter a entidade durante um determinado tempo que pode ser estático ou dinâmico. No caso de tempo dinâmico este muda para cada entrada de entidade e pode ser calculado por funções estocásticas.

O hold tem como função bloquear a entidade que assim vê interrompido o seu percurso no fluxograma. Se o próximo objecto do fluxograma for o *delay*, o *hold* bloqueia a entidade durante o tempo em que outra entidade está no *delay* e assim evitar erros durante a simulação. Assim que a entidade sai do *delay* o *hold* recebe informação para deixa passar nova entidade.

O tempo de *delay* pode ser utilizado para simular movimentos ou tempos de produção durante a simulação.

Os eventos associados a estes objectos são o *on enter* e *on exite* explicados anteriormente.

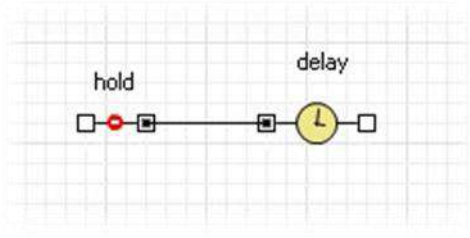

Figura 12- Componentes *hold* e *delay*

Selecte Output – este objecto recebe uma entidade na sua porta de entrada e reencaminha-a para uma das duas portas de saída, esta escolha da porta de saída é feita por uma probabilidade ou por uma condição de verdadeiro ou falso dada a cada porta.

Os eventos associados a este objecto são o *On enter, On exit (true) e On exit (false).*

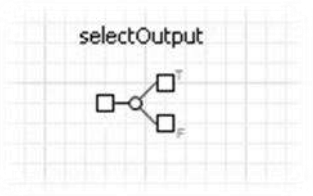

Figura 13- Componente *selectOutput*

3.5 – Máquina de estados.

Utilizadas nos vários active object do trabalho, a máquina de estados é composta por estados e transições. As transições fazem o modelo passar para o estado seguinte e são disparadas no AnyLogic das seguintes maneiras.

- *Timeout* - tempo em que a entidade permanece no estado. Este tempo pode ser estático ou dinâmico, isto é, colocamos directamente um valor do tempo que entidade fica retida no estado ou podemos chamar uma função que vai passar esse tempo de retenção e que varia conforme a situação.
- *Rate* – a passagem para o estado seguinte é definida por uma taxa.
- Condição – a passagem para o estado seguinte é feita por uma condição booleana, esta condição pode ser invocado por uma função que é comparada com a condição de transição.
- Mensagem – A transição é activa quando recebe uma mensagem igual à definida na transição.

No AnyLogic a construção de statecharts está associada à biblioteca *Model*, nesta biblioteca podemos encontrar os objectos *state* e *transition* ilustrados na figura 12.

O *statechart* é usado para dar uma hierarquia aos vários estados para que sejam percorridos de uma maneira ordenada e a sua disposição gráfica é de fácil identificação sendo possível durante a simulação ver qual o estado activo e a activação das transições.

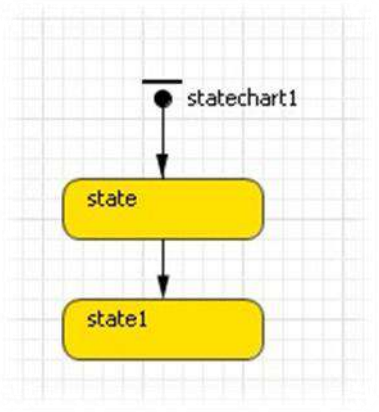

Figura 14- Máquina de estados

Capitulo 4

4. Apresentação e modelagem do primeiro modelo de teste.

Neste capítulo é apresentado o modelo criado de simulação que tem como principal objectivo conhecer a ferramenta utilizada e sua familiarização em sistemas de produção *lean*, *AnyLogic*.

4.1 Introdução

O trabalho tem como objectivo a utilização da ferramenta de simulação AnyLogic para desenvolver um modelo de produção baseado na filosofia lean que interage com outros sistemas implementados na unidade fabril.

O departamento em estudo é o de lixamento e polimento. Este departamento tem dois conjuntos de máquinas que funcionam de maneira autónoma. As peças chegam do departamento de maquinagem são encaminhadas para as máquinas de lixamento onde levam um primeiro tratamento seguindo depois para as máquinas de polimento. Depois de serem tratadas no departamento em estudo seguem para o departamento de galvânica.

No trabalho foram desenvolvidos duas classe Java e dois grupos de Active Object, um grupo de Active Object representa as várias máquinas do polimento e o outro grupo de Active Object representa o lixamento.

- Grupo de Active Object lixamento – usado para simular as máquinas do lixamento;
- Grupo de Active Object Polimento – usado para simular as máquinas de polimento;
- Active Object supermercado – usado para simular armazém onde são colocadas as peças acabadas;
- *Buffer* – representa os *buffers* de entrada de cada máquina;
- Caixa – classe java usada para representar as várias caixas de peças;
- Ordem – representa o número de ordem que cada máquina vai executar:

4.2 Discrição do sistema de produção

O sistema de produção que se pretende implementar está ilustrado na figura 15 e pretende-se que funcione da seguinte maneira.

- Apenas nos interessa os processos de produção do departamento de lixamento polimento
- O tapete vertical é alimentado pelas caixas de peças que saem do supermercado da maquinagem e que depois são distribuídas pelos tapetes horizontais que alimentos as três células de máquinas;
- Cada célula de máquinas é composta por uma máquina de lixamento e uma máquina de polimento;
- Os tapetes horizontais fazem o transporte das peças até aos buffers das máquinas de lixamento depois de tratadas são transportadas para o buffer de polimento e deste para o armazém de peças prontas;
- Cada caixa tem apenas uma referência. A feita a totalidade de cada caixa em cada ordem de produção, logo apenas circulam caixas cheias;
- Quando uma caixa de peças de uma determinada referência chega à máquina de lixamento esta é transferida para o buffer da máquina, quando as peças da caixa acabam a caixa é colocada no tapete e transferida para a máquina de polimento;
- Quando uma caixa chega à máquina de polimento esta é transferida para o seu buffer para ser trabalhada. Depois de todas as peças da caixa serem trabalhadas a caixa é colocada no tapete e transportada para o supermercado onde são guardadas todas as caixas de produto acabado;
- Pretende-se que o fluxo de material em movimento seja mínimo e as cargas de cada máquina sejam equiparadas;
- Os processos de produção devem ser simplificados e o mais autónomos possível para que as várias máquinas possam laborar de modo a independente umas das outras e assim evitar paragem de máquinas por avarias em outras máquinas.

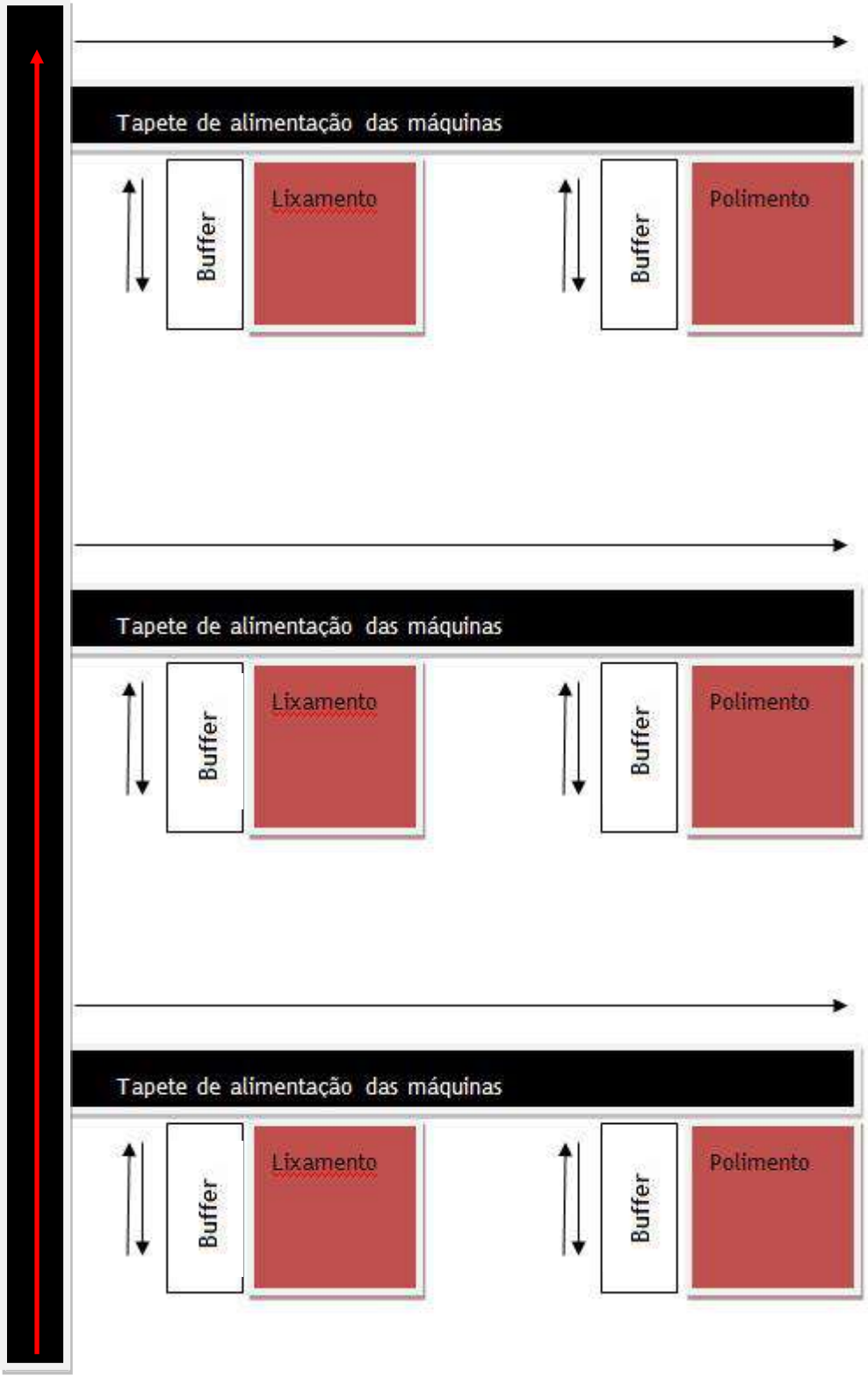

Figura 15 -Sistema de produção idealizado

4.3 Classes java

Foram criadas duas classes java, Caixa e ordem, estas classes java são subclasses da classe principal Entity para que possa ter todos os atributos desta e assim possa percorrer todos os objectos do trabalho. A figura 15 ilustra a classe Principal Entity e as suas subclasses.

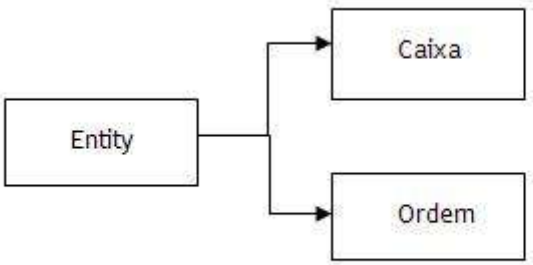

Figura 16 - Classe Entity e suas subclasses

4.3.1 Caixa

A classe Caixa foi criada com o objectivo de simular caixas de peças.

Estas peças são tratadas nas operações de lixamento e polimento e passam pelos vários activemos objects.

A tabela 1 descreve os atributos e características da classe Caixa.

Tabela 1- Atributos da classe Caixa

Nome	**Tipo**	**Descrição**
Ref	Int	Indica a referência das peças que a caixa contém
quantidade	int	Quantidade de peças na caixa

Na figura 16 podemos ver como a classe Caixa é declarada.

Quando é criada uma classe no AnyLogic esta vem logo estruturada por defeito e o utilizador apenas tem de escolher os seus atributos.

A função *toString*() é sobreposta através da instrução *overrride*.

Este método retorna os valores actuais dos atributos da classe Caixa.

Caixa.java

```
/**
 *  Caixa
 */
public class Caixa extends Entity {

      int ref ;

     int quantidade;


    /**
     * Default constructor
     */
    public Caixa(){
    }

    /**
     * Constructor initializing the fields
     */
    public Caixa(int ref, int quantidade){
        this.ref = ref;
        this.quantidade = quantidade;

    }

    @Override
    public String toString() {
        return
            "ref = " + ref +" " +
            "quantidade = " + quantidade +" ";
    }


}
```

Figura 17- Classe java Caixa

4.3.2 Ordem

A classe java Ordem foi criada para simular as ordens de produção que são executadas pelos diferentes Active Objects que representam as máquinas departamento em estudo.

A tabela 2 ilustra os atributos da classe java Ordem.

Tabela 2- Atributos da classe Ordem

Nome	Tipo	Descrição
ref	int	Indica qual a referência a ser executada pela máquina

A figura 17 ilustra a classe Ordem, esta entidade percorre os objectos que simulam as máquinas de polir e de lixar com a ordem a ser executada por cada máquina.

Ordem.java

```
/**
 *  Ordem
 */
public class Ordem extends Entity {

    int ref;



    /**
     * Default constructor
     */
    public Ordem(){
    }

    /**
     * Constructor initializing the fields
     */
    public Ordem(int ref){
        this.ref = ref;

    }

    @Override
    public String toString() {
        return
            "ref = " + ref ;

    }

}
```

Figura 18 - Classe java Ordem

Esta classe é muito parecida com a classe Caixa diferindo apenas no tipo de atributos. Também esta classe utiliza a função *toString*() para retornar os valores actuais dos atributos da classe Ordem.

4.4 Active Objects Criados

Todos ao *Active Object* criados são compostos por parâmetros, variáveis e função. Os parâmetros por norma são constantes e servem para ligar o *Active Object* a outros *Active Object* ou para determinar o estado do próprio.

As variáveis são utilizadas para modelar o *active object* de modo dinâmico já que podem alterar o seu valor no decorrer da simulação e assim alterar o estado do objecto.

Deste modo o uso de parâmetros serve para modelar o comportamento do modelo e o uso de variáveis quando se pretende guardar valores que alteram no decorrer da simulação.

Existe também a *collection Variable* que é utilizada para guardar listas de variáveis.

As funções existentes nos *Active Object* servem para criar código e são chamadas no correr da simulação para modelar o comportamento do modelo, assim quando o modelo está num determinado estado chama uma função que determina o que faz nesse estado e poderá activar uma transição para passar para o estado seguinte.

Os parâmetros são constantes mas têm uma opção em que podem mudar de valor sempre que são chamados se activada o opção dinâmica.

A figura 18 ilustra um parâmetro que está no lixamento 1 e que faz a ligação a outro *active object*, neste caso o polimento1

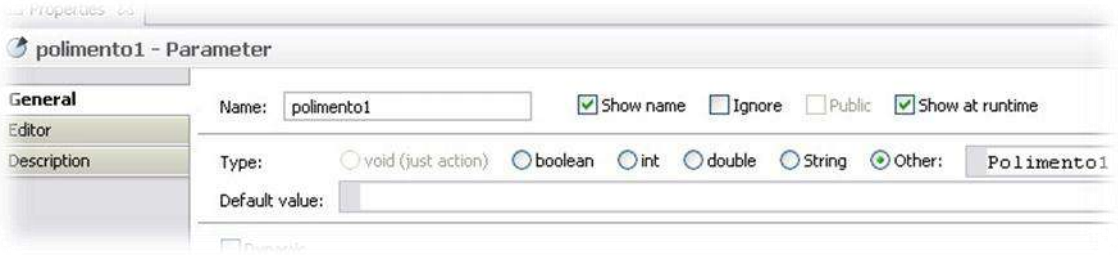

Figura 19 - Parâmetro de ligamento a outro active object

Na figura 19 podemos ver um parâmetro dinâmico em que cada vez que é chamado é gerado um valor aleatório dado pela função triangular, esse valor tem como valor máximo 6, mínimo 2 e valor médio 4.

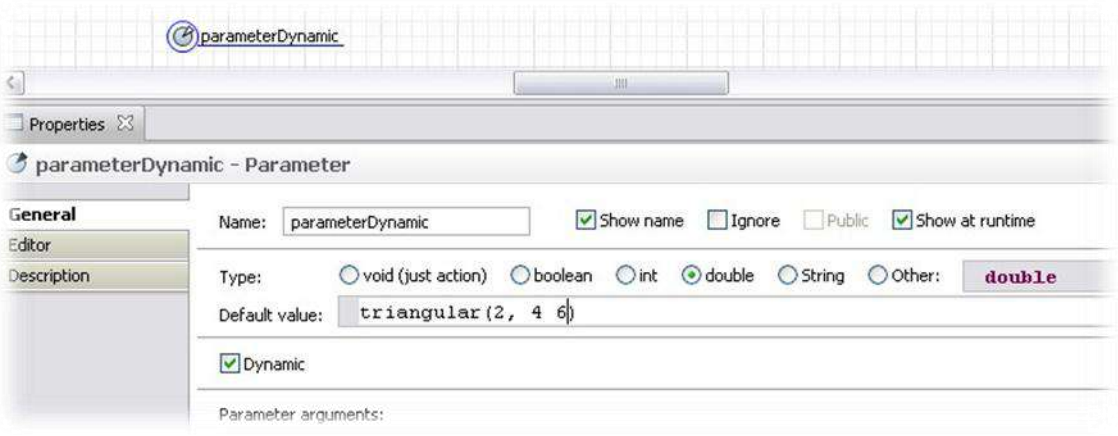

Figura 20- parâmetro dinâmico

A diferença entre os dois métodos de atribuir valor a um parâmetro é que quando este não é dinâmico o seu valor é sempre o mesmo e quando é dinâmico o seu valor varia de cada vez que é chamado.

A criação dos vários *active Object* serve para simular as várias máquinas do departamento em estudo sendo o buffer uma elemento importante pois entra nos vários *active object*.

4.4.1 Buffer

O *buffer* tem como função armazenar as entidades que representam as caixas das várias referências. Para que as entidades possam entrar no *buffer* é necessário um *enter* e para que saiam é necessário o método *remove* sempre que as entidades não seguem o caminho normal do fluxograma.

O que representa o buffer é o componente *queue* existente na *Entreprise Library* e tem os atributos capacidade que indica qual a quantidade de caixas capaz de armazenar e o tamanho que indica o número de caixas presentes no *buffer*.

4.4.2 Supermercado

O supermercado é onde são armazenadas as caixas de peças, logo está associado a *buffers* em que cada *buffer* guarda apenas caixas de uma determinada referência.

O supermercado terá tantos *buffers* quantas as referencia a produzir.

O supermercado está ligado aos três active object polimento pelos parâmetros respectivamente polimento1, polimento2 e polimento3. São estes *active object* que colocam as caixas com as peças acabadas no respectivo *buffer* associado à referencia da peça no supermercado.

A criação da caixa de peças de determinada referência no supermercado feita através do componente *source* da *Entreprise Library* em que faz a representação visual da caixa feita num dos três *active object* polimento.

O supermercado tem uma função chamada RetiraSuperm que vai simular a retirada de caixas do supermercado quando tem uma quantidade superior a seis caixas em qualquer um dos *buffers*.

A figura 21 ilustra a criação das caixas prontas no respectivo *buffer* do supermercado.

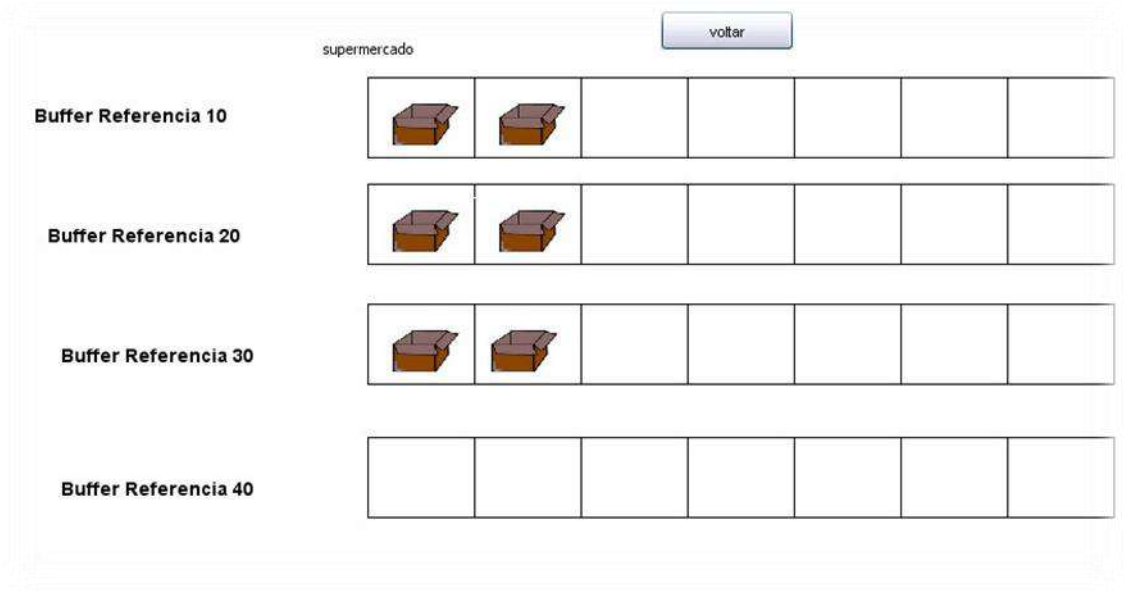

Figura 21 - Supermercado

4.4.3 Polimento

As máquinas do polimento são representadas por três *active objects*, respectivamente polimento1, polimento2 e poliemnto3.

Embora funcionem de modo autónomo e completamente independentes uns dos outros, estes *active object* são muito parecidas e a descrição do funcionamento de um vale para todos.

É este *active object* que recebe as ordens de produção que estão numa base de dados e são transferidas para uma *collectionVar* de nome ordensPolimento1 referente ao polimento1.

Associado a cada referencia está um *buffer* de entrada que recebe as caixas de peças tratadas no lixamento e que serão polidas aqui. Depois de polidas estas caixas de peças são colocadas no supermercado.

Na figura 22 está representado o polimento que tem ligação ao supermercado e ao lixamento através de parametros de ligação.

As variáveis actual e ref guardam respectivamente a ordem actual e a referencia que está a ser produzida.

As função representadas no quadro à esquerda são chamadas em várias etapas da simulação e em determinados estados da seguinte maneira.

- RecebeOrdem – recebe a ordem que está no *array* de variáveis ordensPolimento e coloca a referencia a ser produzida na classe java ordem
- Verificacx – função booleana que verifica qual a referencia a produzir e se existem caixas no respectivo *buffer* com essa referencia. Se retornar *true* passa para o estado verificaRef senão fica no estado FaltaCX em espera que entre uma caixa dessa referencia no respectivo *buffer*.

- Needsetup – compara a ordem actual com a proxima ordem. Se forem diferentes entra no estado *SetUp* senão passa para o estado Trabalho.
- Cxpronta – esta função é chamada no estado Trabalho em que quando a entidade sai do referido estado coloca a caixa pronta no respectivo *buffer* do supermercado e cria uma ordem de fabrico para o lixamento da referencia produzida para que a caixa de peças que saiu seja reposta.
- bufferEntrada – função auxiliar que é chamada por outras funções. Serve para retornar o *buffer* da referencia a produzir.
- tempoSetup – função que determina o tempo que cada referencia precisa para fazer o *setup*.
- tempoOperacao - função que determina o tempo que cada referencia precisa para ser produzida.

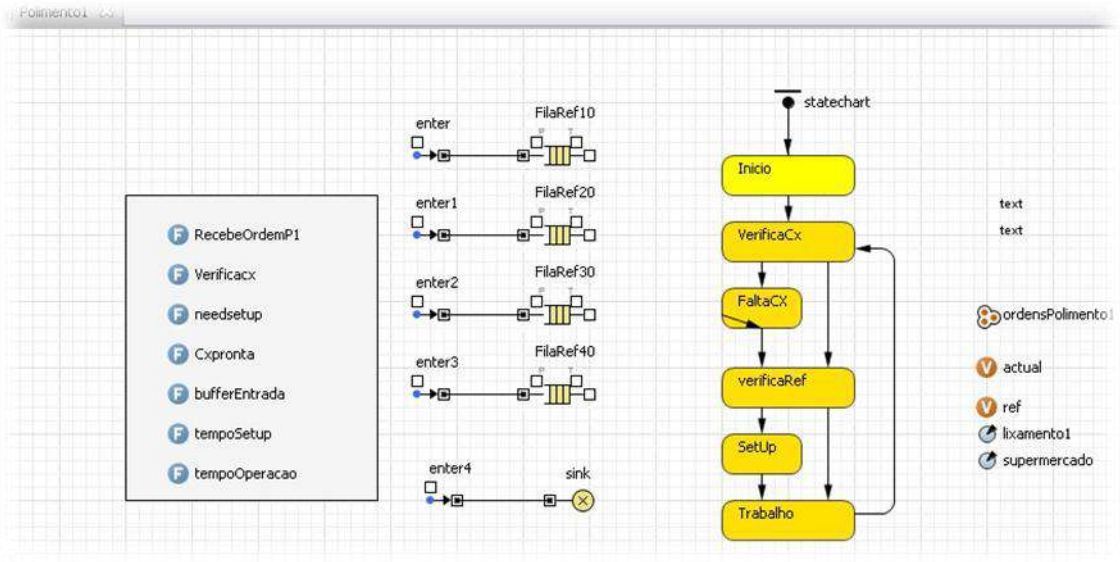

Figura 22 – Polimento

4.4.4 Lixamento

As máquinas do Lixamento são representadas por três *active objects*, respectivamente lixamneto1, lixamneto2 e lixamneto3.

Embora funcionem de modo autónomo e completamente independentes uns dos outros, estes *active object* são muito parecidas e a descrição do funcionamento de um vale para todos.

Este *active object* tem a função de repor as caixas consumidas no polimento, isto é, sempre que uma caixa é consumida no polimento é gerado uma ordem de produção entregue no lixamento para que essa caixa de peças seja reposta.

A ordem de produção é guardada numa *collectionVar* de nome ordensLixamento1 referente ao lixamento1.

Associado a cada referencia está um *buffer* de entrada que recebe as caixas de peças do departamento de maquinagem e que serão lixadas nesta máquina. Depois de lixadas estas caixas de peças são colocadas no *buffer* da respectiva referencia da máquina de polimeto.

Na figura 23 está representado o lixamento que tem ligação ao polimento através do parametro de ligação.

As variáveis actual e ref guardam respectivamente a ordem actual e a referencia que está a ser produzida.

As função representadas no quadro à direita são chamadas em várias etapas da simulação e em determinados estados da seguinte forma:

- RecebeOrdem – recebe a ordem que está no *array* de variáveis ordenslixamento1 e coloca a referencia a ser produzida na classe java ordem
- Needsetup – esta função controla a transição para o estado seguinte no estado verificaRef através da comparação da ordem actual com a proxima ordem. Se forem diferentes entra no estado *SetUp* senão passa para o estado Trabalho.
- Abastece – esta função vai reabastecer o *buffer* da referencia a produzir, isto é, verifica qual a referencia de caixa de peças a produzir e vai retirar essa caixa ao supermencado da maquinagem e colocar no respectivo *buffer* da máquina de lixamento.
- ColocaPoli – esta função é chamada quando a entidade sai do estado Trabalho em que é feita uma nova caixa de peças, esta caixa de peças é retirada do lixamento e colocada no respectivo *buffer* da máquila de polimento.
- tempoSetup – função que determina o tempo que cada referencia precisa para fazer o *setup*.
- tempoOperacao - função que determina o tempo que cada referencia precisa para ser produzida.

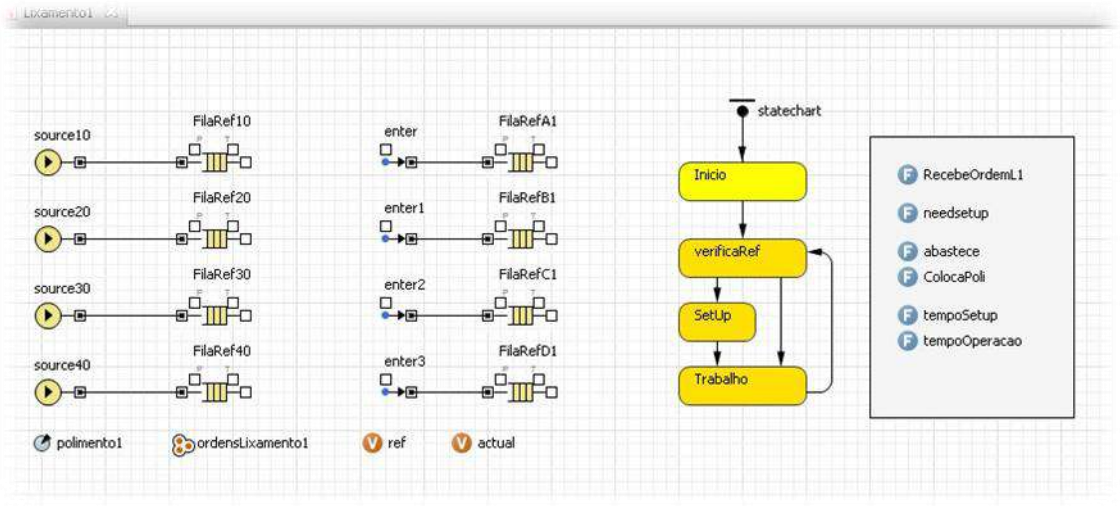

Figura 23- Lixamento

O supermercado da maquinagem é representado pelas *queue* e *source* à esquerda.

Este supermercado não é tratado no trabalho já que pertence a outro departamento mas está representado na simulação como ilustra a figura 24.

Como se pode ver a cada *buffer* estão associadas caixa de 20 peças da respectiva referência. Na ilustração o *buffer* da referência 10 tem 4 caixas, o *buffer* da referencia 20 tem 6 caixas, o *buffer* da referência 30 tem 8 caixas e o *buffer* da referencia 40 tem 4 caixas.

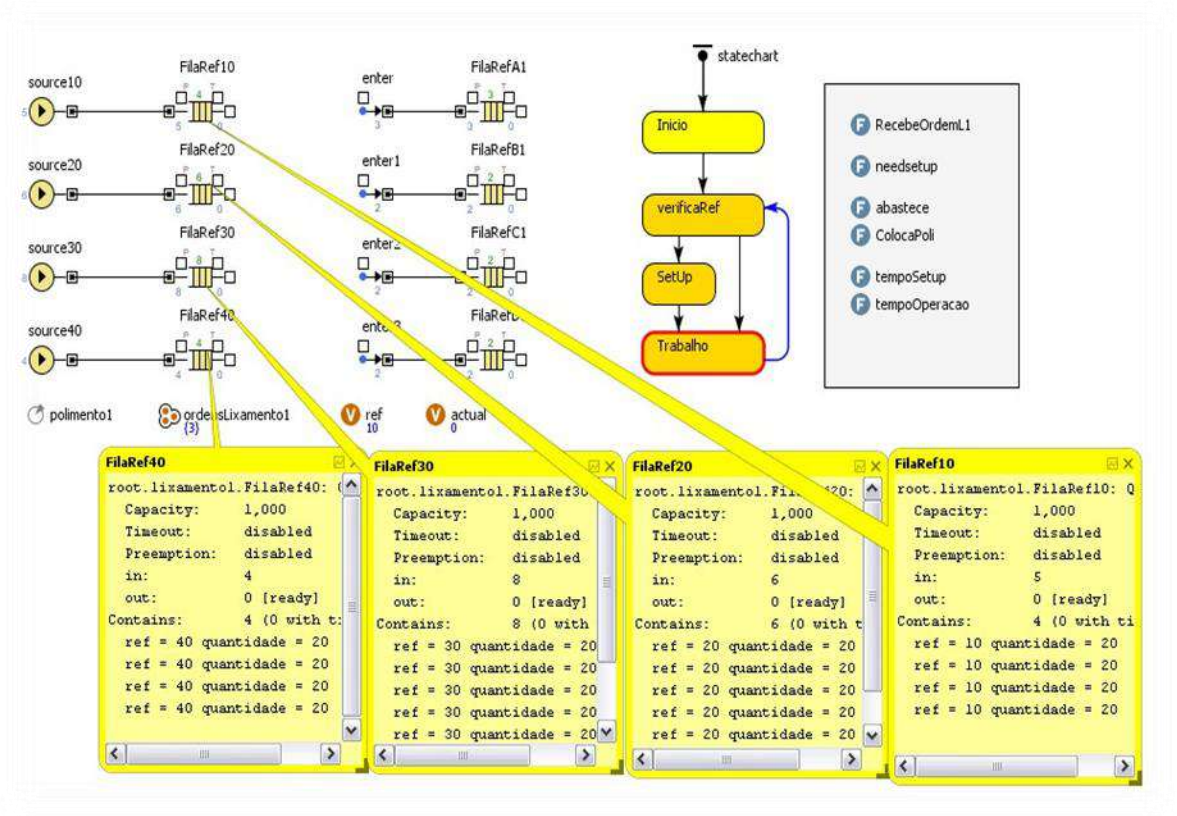

Figura 24 - Supermercado relativo ao departamento de maquinagem

4.4.5 Main

A figura 25 ilustra o *main*, este é o *active object* principal e faz a ligação com todos os outros. Esta ligação é feita arrastando cada *active object* para o editor gráfico do *main*, como se pode ver estou representados o lixamento1, lixamento2, polimento1, polimento2, os outros não aparecem na figura 25 por não conseguir colocar todos.

O componente *database* faz a ligação entre o *AnyLogic* e a base de dados onde se encontram as ordens de produção.

A função ordens serve para colocar todas as ordens na simulação e assim poderem ser acedidas quando se clica no botão Ordens de produção.

O botão supermercado faz a ligação à representação gráfica das caixas a entrarem no supermercado.

Do lado direito de cada máquina tem o buffer da respectiva referência onde são colocadas as caixas de peças para produção. Esta imagem nas máquinas representa que estão a trabalhar, quando estão paradas ou em *setup* a imagem muda.

O texto do lado direito de cada máquina mostra a referencia que está a ser produzida e a próxima referência a produzir.

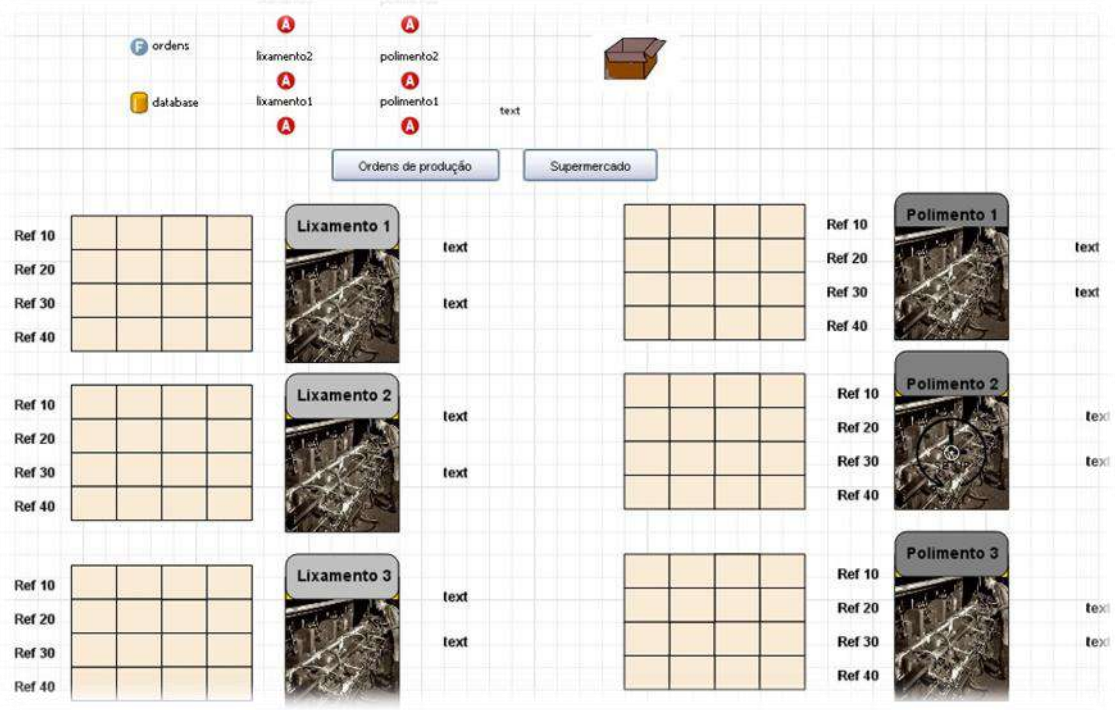

Figura 25 – *Main* do modelo de simulação

4.5 Dimensionamento dos buffers e afectação das máquinas

O pretendido com este modelo de simulação seria aplicar os conceitos da filosofia *lean* em que as máquinas funcionam sem avarias e onde os fluxos e cargas estão todos nivelados em que as máquinas funcionam para repor o stock e que este seja mínimo mas de modo a que não ocorram paragem das máquinas por falta de *stock*.

O trabalho foi evoluindo para que as máquinas de lixamento e polimento trabalhem com taxa máxima desde que o *buffer* da referência produzida não esteja cheio. Para isso foram sendo introduzidas alterações para que os buffers fossem o mínimo possível e que as máquinas estivessem sempre a trabalhar durante a simulação e nunca tivessem ruptura de *stocks*.

As máquinas de polimento produzem as referências de semi-acabados para alimentar o armazém da galvânica e quando consumem um produto de uma determinada referência geram uma ordem para o lixamento para que a caixa que saiu seja reposta.

Para que as máquinas de polimento possam logo começar a trabalhar foram colocadas duas caixas de cada referencia nos respectivos *buffer*. A ordem de produção dada ao lixamento é lançada quando o polimento acabar de fazer a caixa em produção mas com o tempo de produção e com os tempos de setups as máquinas de lixamento ficavam paradas, por isso foram colocadas duas ordens de produção em cada máquina de lixamento para que esta possa trabalhar sem ter de ficar à espera da ordem do polimento o que provocava paragem destas e por consequência também iria ter falta de caixas na máquinas de polimento pois não estava a ser alimentada.

A figura 26 ilustra o *startup* do lixamento com duas ordens de produção inseridas quando a simulação arranca.

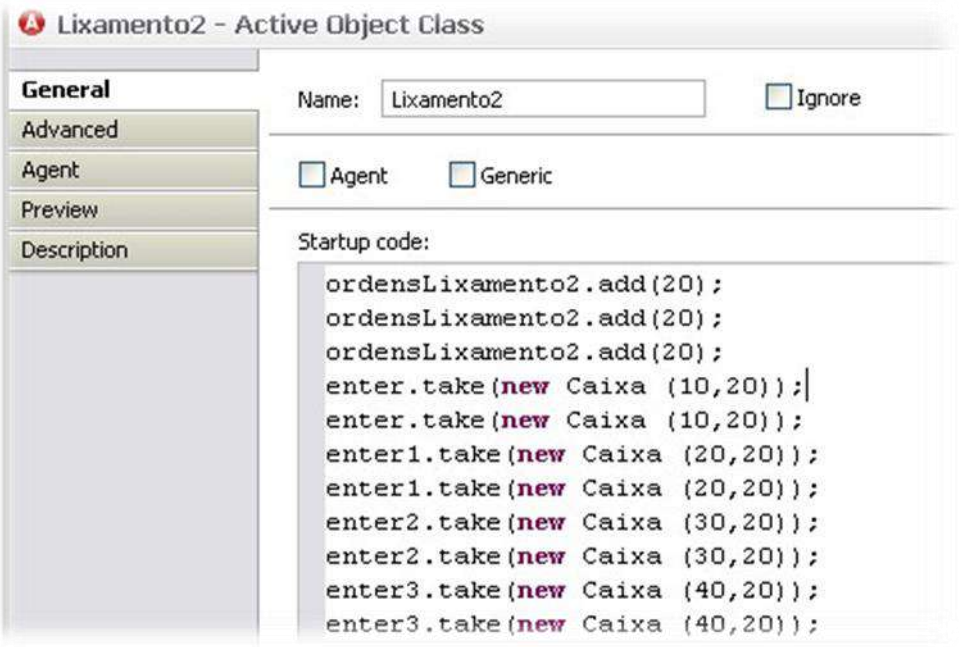

Figura 26- *Starup* do lixamento.

A colocação de duas caixas de cada referência nos *buffers* de cada máquina do lixamento foi escolhida de moda a que os buffers fossem mínimos e sem ruptura de stock, já os *buffers* do polimento foram inicializados com três caixas de cada referência.

A figura 27 ilustra a situação de extremo em que o *buffer* da referência 10 da máquina de polimento 2 fica sem caixas mas a máquina lixamento 2 que abastece esse buffer vai mudar para referência 10 de modo a abastecer esse buffer e não ocorra ruptura quando o polimento 2 voltar a fazer a referência 10 que neste momento está a produzir a referência 20 que está com o *buffer* de entrada cheio. O mesmo acontece com o polimento 3 em relação ao buffer da referência 30.

Também é possível observar que o *buffer* de referência 10 do polimento 1 se encontra cheio mas a máquina já está a produzir essas referências e o nível do buffer vai baixar visto que o lixamento 1 está a produzir a referência 40 para alimentar o buffer de entrada de referência 40 do polimento 1 que é o que está mais baixo.

O mesmo acontece para as outras máquinas do lixamento em que estas vão alimentar os *buffers* de entrada das máquinas de lixamento que se encontram com os níveis mais baixos.

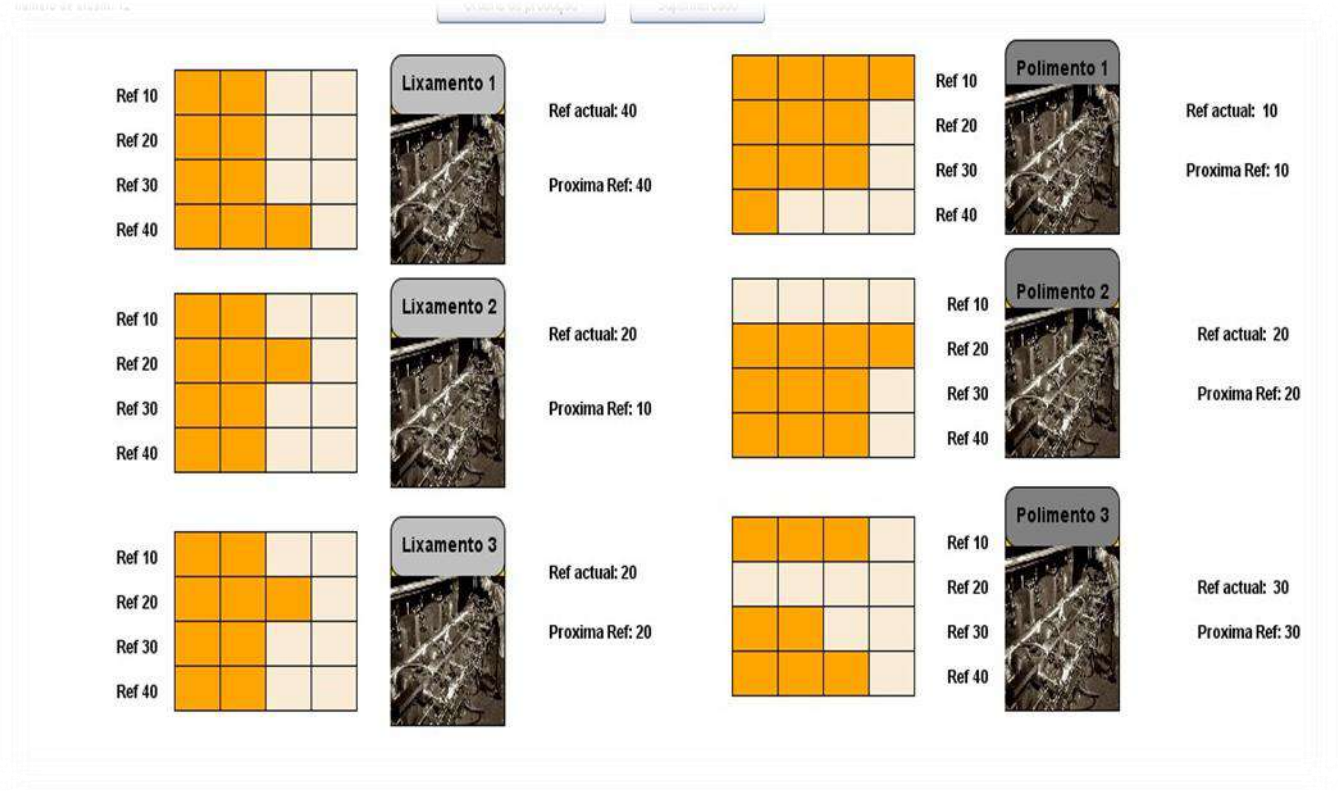

Figura 27 – Simulação.

Capitulo 5

5. Simulação de vários cenários de procura e tempos de setup

Neste capítulo é desenvolvido um novo modelo de simulação com introdução de variáveis que alteram a produção planeada tais como: variação da procura, as avarias das máquinas e quando uma máquina trabalha com mais que uma referência os tempos de *setup*.

5.1 Introdução

Como referido no capítulo 1 o trabalho tem como base de estudo uma empresa de metalomecânica que se dedica ao fabrico de torneiras.

A empresa fabrica uma vasta gama de produtos com volumes de produção semanais muito variáveis.

Em anos recentes a empresa tem feito um esforço para evoluir no sentido de uma produção *lean* no entanto tem se deparado com dificuldades ao tentar criar fluxos *pull* porque o seu sistema de produção está sujeito a múltiplos factores de instabilidade entre os quais taxas de avarias de equipamentos, tempos de *setup* elevados e com grande variabilidade e grandes flutuações na procura.

Para alguns produtos em que a procura está relativamente estabilizada foi possível evoluir para fluxos puxados.

Assim uma questão que se coloca aos responsáveis da produção da empresa é saber que produtos devem trabalhar em *pull* e que produtos devem funcionar em *puhs* e assim decidir a afectação das máquinas. Os produtos com uma procura estabilizada podem ser produzidos por máquinas dedicadas e os produtos com uma procura muito variada podem ser produzidos por várias máquinas e assim a mesma máquina pode trabalhar com várias referências e aproveitar os tempos de menor procura para produzir outras referências em que os stocks sejam reduzidos.

Pretende-se saber a partir de que procura as máquinas deixam de ser dedicadas a apenas uma referência e passam a fazer várias referências. Esta resposta depende do tempo de setup das

máquinas, pois quanto maior for o tempo de setup da máquina para mudar de referência menos vezes o poderá fazer porque a máquina estaria muito tempo parada em *setup*.

Destas decisões vai depender o dimensionamento dos *buffers* de cada referência pertencentes ao supermercado à saída do polimento que contém os produtos semi-acabados.

5.2 Modelação de cenários

Para a modelação dos cenários foram criados dois modelos de simulação, um mais simples com apenas duas máquinas e um supermercado, e outro mais complicado em que cada máquina pode produzir várias referências.

No primeiro modelo as máquinas funcionam de modo independente e dedicadas a apenas uma referência, o supermercado recebe as peças produzidas por cada uma das máquinas.

Foram criados três *active Object*, dois para representar cada uma das máquinas e um para representar o supermercado com buffers para cada uma das referências.

A figura 28 ilustra um *active object* de uma máquina que é constituído por dois gráficos de estados, um representa a produção e outro representa as avarias.

Quando ocorre uma avaria a máquina principal é colocada no estado parado.

A função CXfeitas é chamada à saída do estado trabalhar do gráfico de estados e coloca a caixa produzida no buffer da sua referência existente no supermercado.

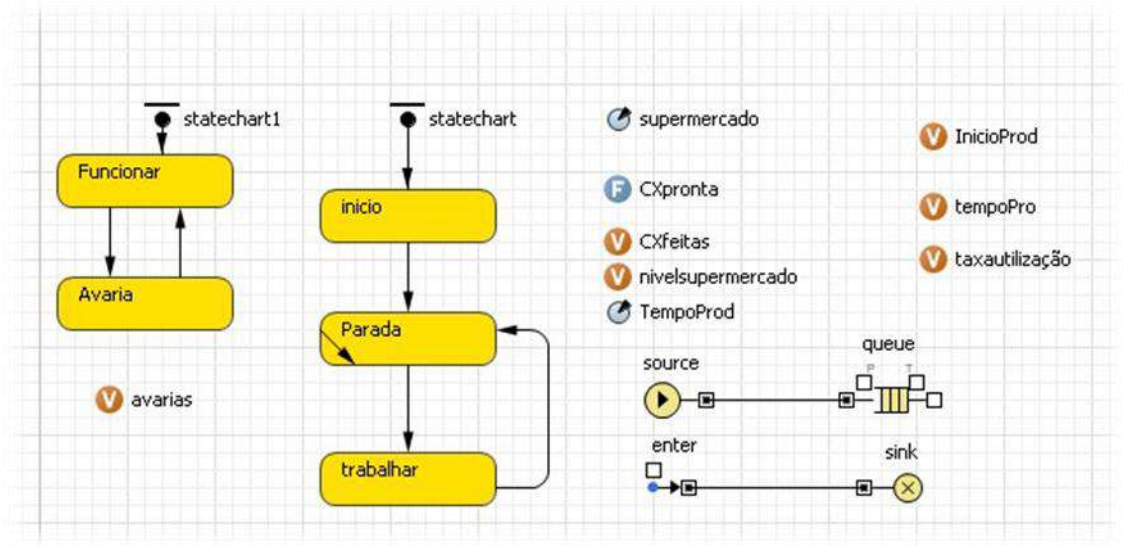

Figura 28 - Active object da máquina

No supermercado tem também dois gráficos de estados para representar a procura das respectivas referências ao longo de cada dia da semana.

Na figura 29 podemos ver a ilustração do supermercado utilizado para guardar as peças que saem do polimento e para simular a saída de peças que representa a procura ao longo da semana.

Os dias na parte de cima da figura 29 são configurados com o tempo de ciclo do gráfico de estados que representam a procura em cada dia da semana.

As funções calculaA e calculaB fazem o controle do tempo de simulação em que para cada dia é indicado o tempo de ciclo do gráfico de estados que simula a procura.

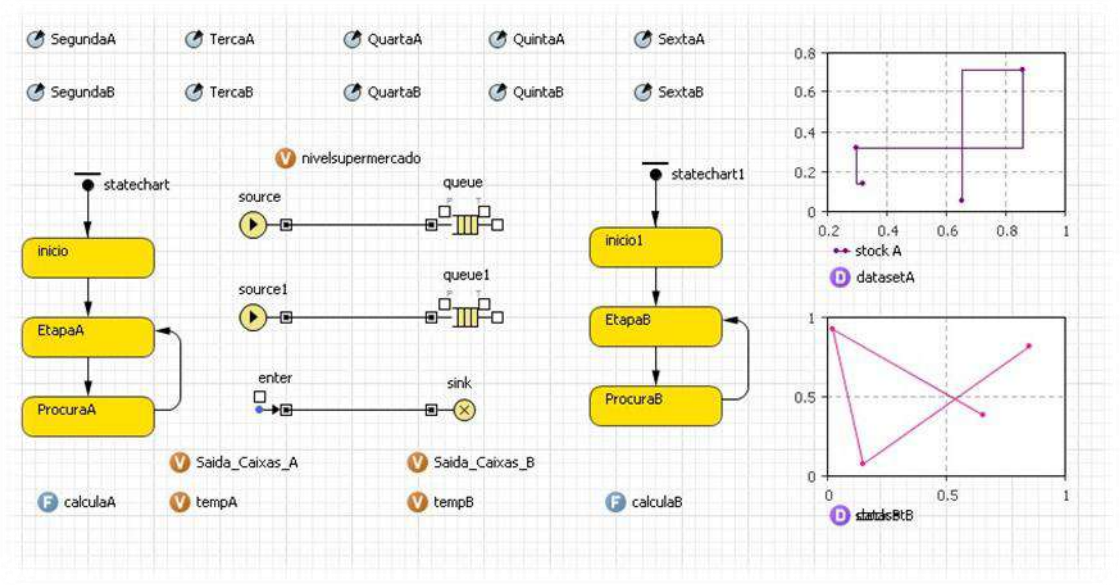

Figura 29 - Supermercado.

No segundo modelo as máquinas podem funcionar de modo dedicado a apenas uma referência ou podem produzir várias referências, adaptando a produção às referências com maior procura cujos *stocks* estejam mais baixos.

Assim, no segundo modelo são acrescentadas duas novas máquinas ficando com duas máquinas dedicadas a apenas uma referência em que a referência que produz tem uma procura nivelada e próxima da sua capacidade de produção e duas máquinas que produzem várias referências para as situações em que a procura tem grandes oscilações. As máquinas que produzem várias referências verificam qual o *buffer* que tem o nível mais baixo e se a referência que está a produzir tem o nível do *buffer* alto então faz um setup para e passa a produzir essa referência.

A figura 30 ilustra uma máquina que pode fazer mais que uma referência. De notar que o respectivo gráfico de estados tem o estado de *setup* que determina o tempo de *setup* da máquina quando muda a referência a produzir.

Para definir as condições de mudança de produção foram criados dois parâmetros, *red* e *green que* indicam os limites críticos dos *buffers* das referências que a máquina pode produzir. Sempre que o nível do *buffer* da referência em produção está acima de *green* à máquina mudar de referência se um dos buffers das outras referências que pode produzir está abaixo de *red*.

A função control verifica se houve mudanças de referência durante a simulação.

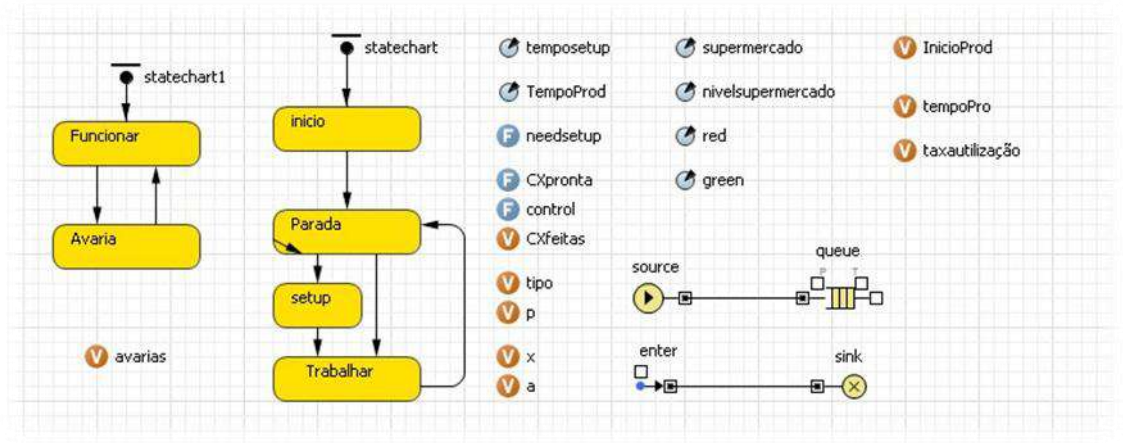

Figura 30 - Máquina que pode fazer várias referências.

5.3 Criação de cenários.

Dado o grande número de referências, equipamentos e variada procura a estratégia foi criar um conjunto de cenários idealizados mas mesmo assim suficientemente próximos da realidade que permitam obter uma primeira validação do sistema de produção.

Como as variáveis são várias e aleatórias, vamos considerar cenários para três factores determinantes no sistema de produção da empresa, nomeadamente: avarias das máquinas, tempos de *setup* e variação da procura.

O departamento que funciona em célula é aqui considerado como apenas uma máquina com um supermercado à saída onde são colocados os produtos semi-acabados. O supermercado tem um *buffer* para cada referência.

A procura quando nivelada é de 100 peças por dia o que equivale a uma caixa por hora de 10 peças para um dia de trabalho de 10 horas.

As ferramentas desenvolvidas permitem ao gestor efectuar análises de sensibilidade relativamente a factores como o tempo de *setup*, variação da procura, e avarias das máquinas e avaliar o seu impacto sobre o dimensionamento dos *buffers* à saída do polimento.

De modo a normalizar os dados são considerados cinco dias de trabalho semanais em que cada dia de trabalho tem 10 horas com uma procura média diária de 100 peças/dia durante os cinco dias da semana.

5.3.1 Máquina que produz apenas uma referencia com procura nivelada.

O primeiro cenário é o mais simples, uma procura nivelada ao longo dos dias da semana com uma máquina afectada a apenas uma referência e numa primeira fase sem avarias.

A figura 31 ilustra a uma máquina em que a procura está nivelada e é igual à produção.

A taxa de utilização da máquina é de 100% pois não tem paragem.

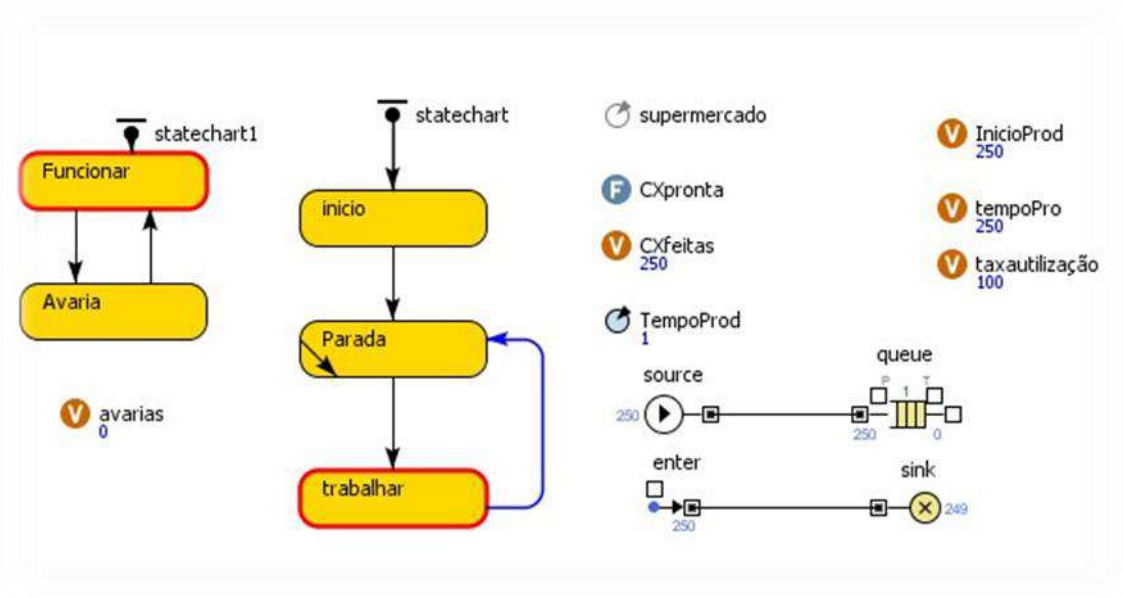

Figura 31 - Máquina afecta a uma referência.

A figura 32 ilustra o supermercado que recebe as referências do polimento. Foram simuladas 250 horas o que corresponde a cinco semanas de 10 horas por dia.

Como se pode ver com um nível de supermercado inicial de duas caixas este nível é sempre mantido, o que era esperado pois as máquinas produzem o mesmo que a procura.

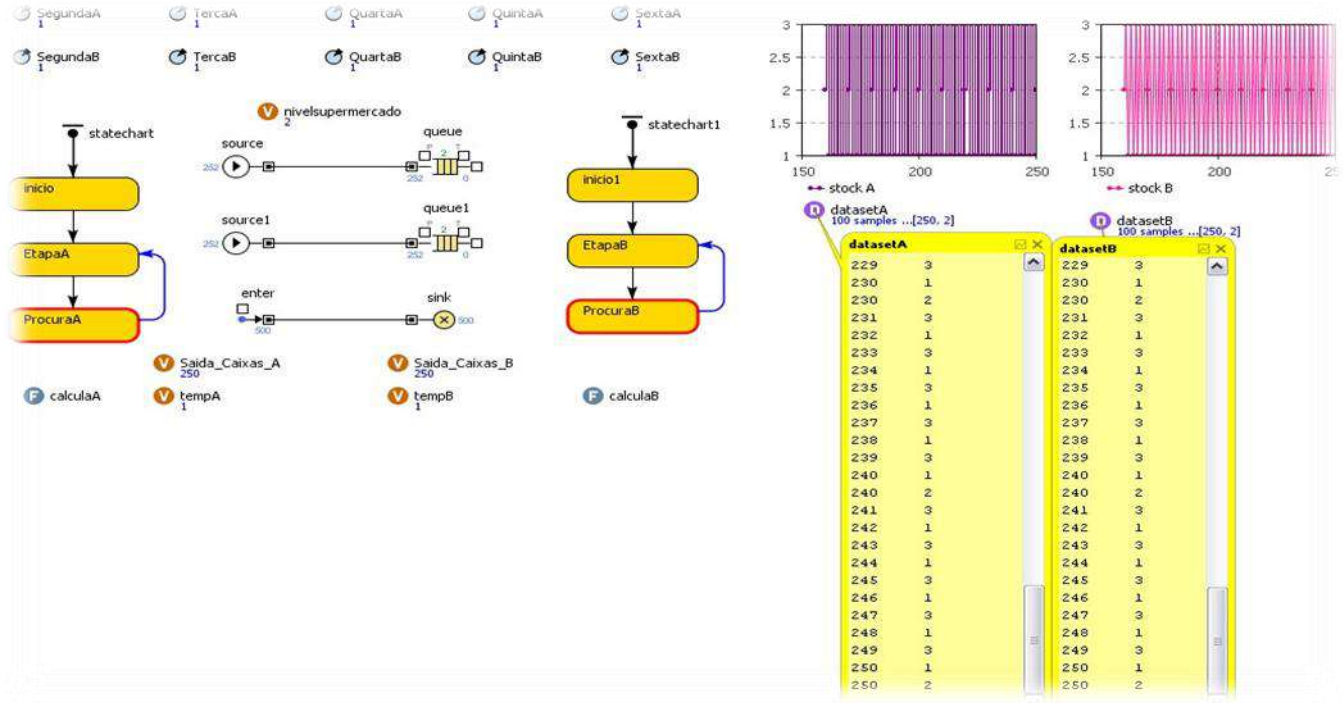

Figura 32 - Supermercado.

5.3.2 Dimensionamento do buffer para procura não nivelada e sem avarias

Para uma simulação de 50 horas, uma semana, com uma procura média de 100 peças por dia.

A variação ao longo de cada semana é ilustrada na figura 33 em que a variação entre dias consecutivos é Δ. O nível dos buffers é inicializado com dez caixas em cada semana de simulação.

Se PN é a procura nominal, vamos alterar o valor de Δ para cada semana de simulação em que este valor varia de 10%. Assim se na primeira semana não tem variação Δ=0%, na segunda semana Δ=10%, foram feitas simulações de várias semanas até uma variação de procura de Δ=40%.

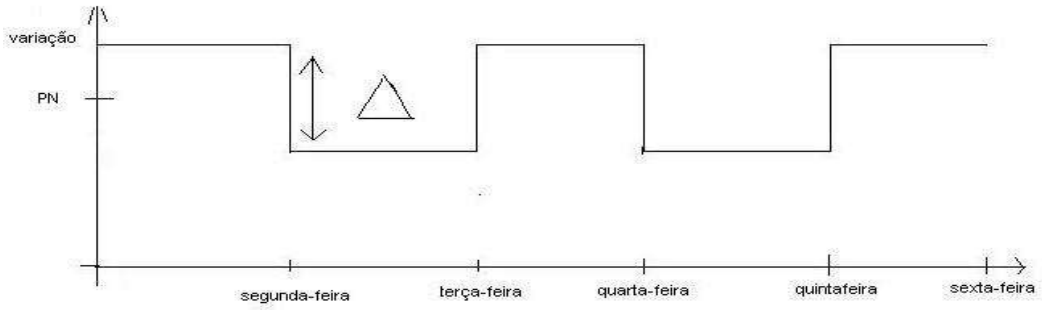

Figura 33 - Variação da procura ao longo da semana.

Como se pode observas na figura 34 para uma variação da procura, Δ=20%, o nível mínimo do *buffer* é de 6 caixas o que indica que o buffer pode ter um nível mínimo inicial de cinco caixas sem que ocorram rupturas.

Neste caso em que Δ=20% a variação semanal de dia para dia é a seguinte: se começa com uma procura de 80% na segunda-feira, na terça-feira essa procura passa para 120% e assim para os restantes dias da semana.

De notar que cada dia da semana tem um tempo de ciclo de procura associado.

Se num dia a procura é de 80% o tempo de ciclo é 1.2, ou seja, demora mais 20% que um ciclo normal a que corresponde menos 20% da procura média.

Para as restantes variações de procura apenas se vai preencher a tabela 3, servindo as figuras 34 e 35 como ilustração de uma variação de procura.

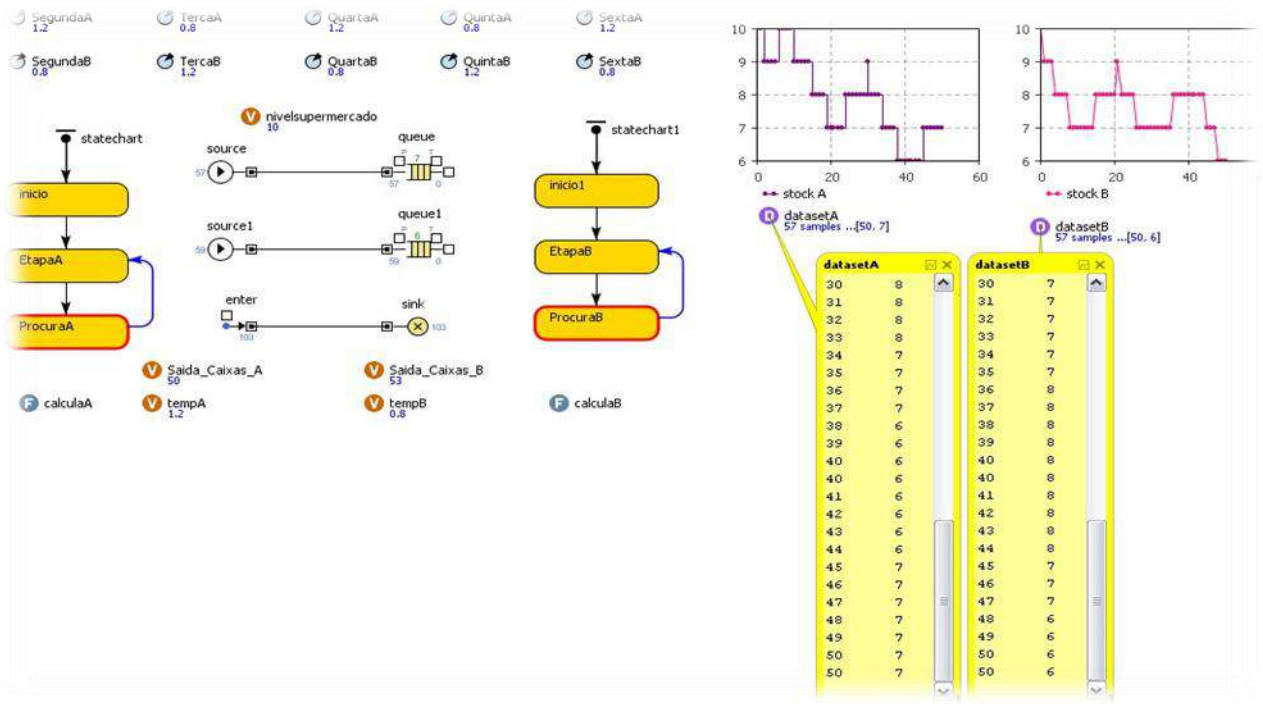

Figura 34- Variação da procura de 20%

Na figura 34 é possível observar uma taxa de utilização da máquina de 95 % para uma variação da procura de 20% de dia para dia ao longo da semana.

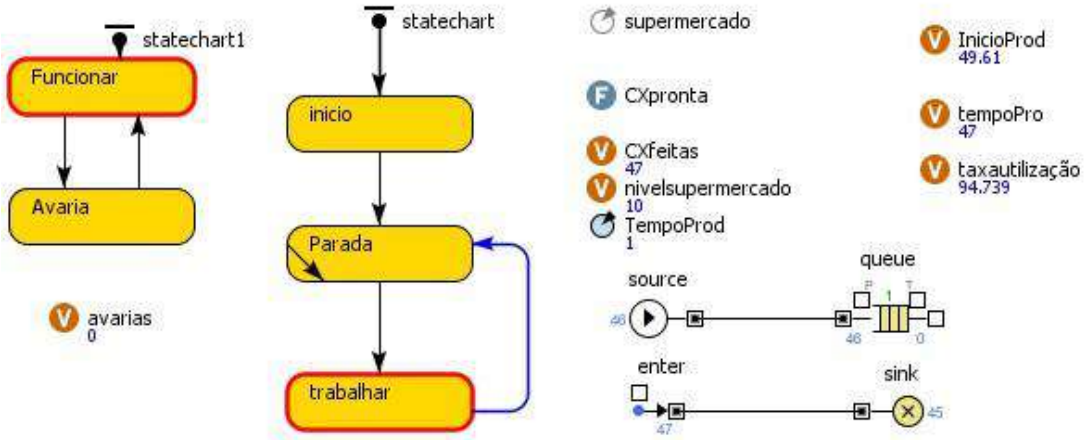

Figura 35 - Taxa de utilização para variação da procura de 20%

Como seria de esperar pela análise da tabela 3 é possível concluir que quanto maior é a variação na procura maiores terão de ser os buffers e menor é a taxa da utilização da máquina.

A taxa de utilização menor é explicada pelo facto da máquina começar com uma procura inicial menor no primeiro dia e como a sua capacidade de produção maior que a procura, fica parada porque o *buffer* está cheio.

Já se a procura do primeiro dia fosse maior do que a produção normal, a máquina teria uma taxa de utilização maior, pois estaria menos tempo parada no inicio.

Na primeira coluna da tabela 3 é apresentada a variação de dia para dia ao longo de cada semana de simulação, na primeira semana não tem variação pois a procura é constante, na segunda semana tem uma variação de 10% de dia para dia ao longo da semana, isto é, no primeiro dia tem uma procura de 90% no segundo de 110% e assim por diante, igualmente para as restantes semanas.

Na coluna 2 o nível inicial dos *buffers* é o nível com que os buffers começam no inicio da simulação de modo a ter uma margem segura de não ruptura. Na coluna 3 tem o mínimo atingido de cada *buffer*, logo a diferença entre o nível inicial e o nível mínimo dá a coluna 4 que é o dimensionamento dos buffers para que seja mínimo e não ocorram rupturas. A fórmula para que não ocorram rupturas do dimensionamento do *buffer* é a seguinte:

Nível mínimo do buffer = Nível inicial – nível mínimo + 1

Tabela 3 - Variação da procura

Variação de dia para dia (%)	**Nível inicial do *buffer***	**Nível mínimo atingido do *buffer***	**Nível mínimo inicial do *buffer***	**Taxa de utilização da máquina (%)**
0	2	1	2	100
10	10	8	3	97,76
20	10	6	5	94,74
30	10	4	7	94,36
40	10	1	10	92,35

5.3.3 Dimensionamento do buffer para procura não nivelada com avarias

Para uma simulação de 50 horas, com uma procura média de 100 peças por dia para uma semana mas com uma variação de dia para dia de 10% em 10% para cada semana e com os buffers no nível máximo no inicio da semana igual a 20 caixas com simulação de 6 avarias por semana em que cada avaria demora uma hora.

Como se pode ver na figura 36 para uma procura com uma variação de 20% em relação à procura normal diária semanal o nível mínimo do *buffer* é de 15 caixas o que indica que o nível mínimo do *buffer* no inicio da simulação pode ser de 6 sem que ocorram rupturas.

Para as restantes variações de procura procuras apenas se vai preencher a tabela 4 como foi feito no exemplo anterior sem avarias.

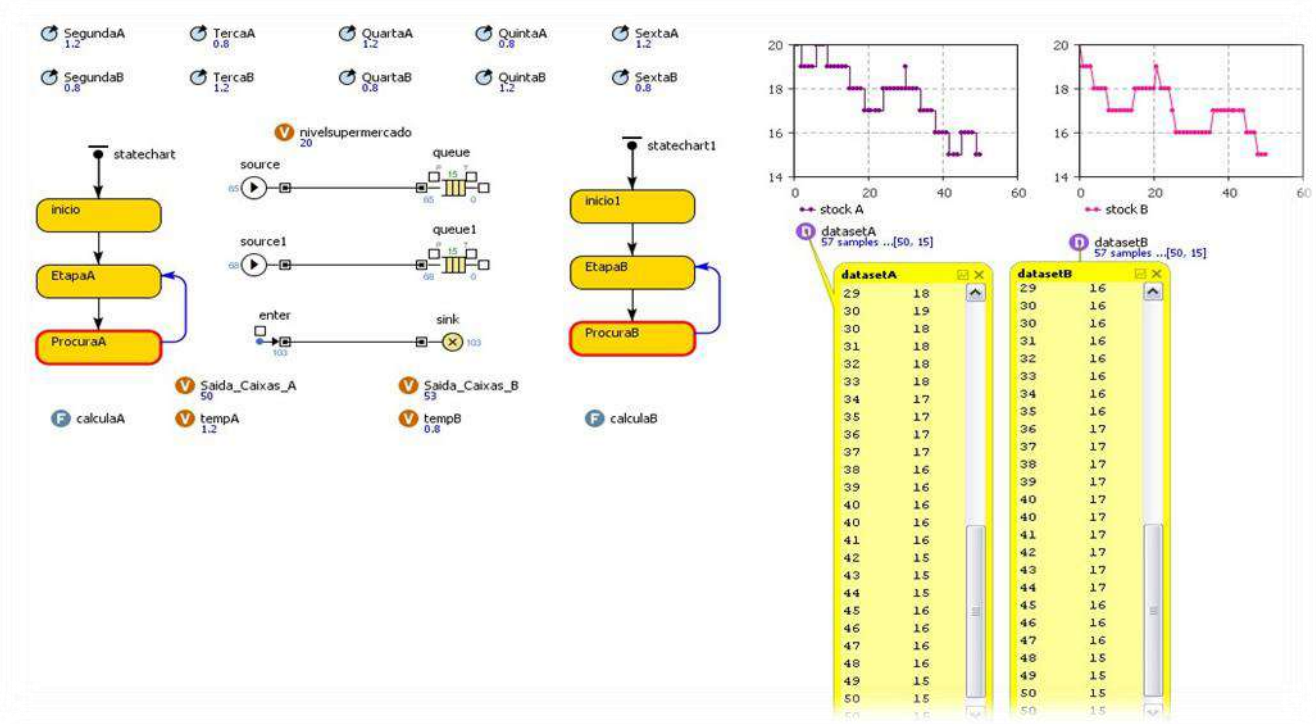

Figura 36- Procura com variação de 20% com avarias

A figura 37 ilustra a simulação com avarias para uma variação de 20% da procura em relação à média em que a maquia funciona com uma taxa de 92%.

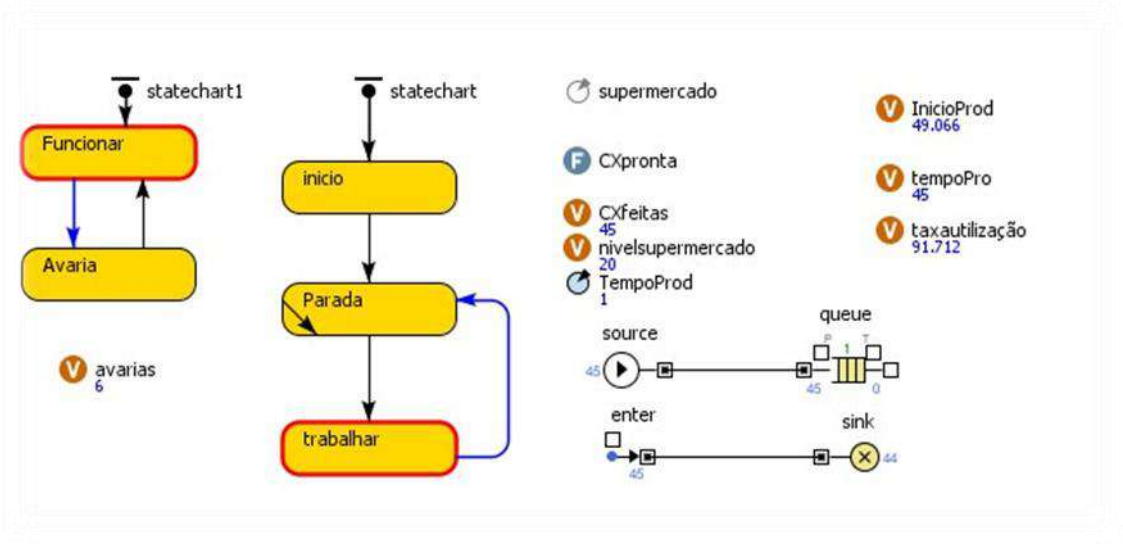

Figura 37 - Taxa de utilização da máquina com variação na procura de 20% e 6 avarias.

Pela análise da tabela 4 é possível concluir que quanto maior é a variação na procura maiores terão de ser os buffers e menor é a taxa da utilização da máquina embora a variação da taxa de utilização da máquina não varie tanto como no caso sem avarias.

Tabela 4 - Variação da procura com avarias.

Variação de dia para dia (%)	Nível inicial dos *buffers*	Nível mínimo atingido do *buffer*	Nível mínimo inicial do *buffer*	Taxa de utilização da máquina (%)
0	20	16	5	93,75
10	20	16	5	92,66
20	20	15	6	91,71
30	20	12	9	90,65
40	20	10	11	90,64

5.4 Criação de cenários para máquinas que podem fazer mais que uma referencia

Quando a procura é muito alterada ao longo da semana e quando as máquinas têm avarias demoradas isso provoca um desequilíbrio no supermercado podendo ocorrer rupturas de stock. Para não ter rupturas os stocks têm de ser muito grande para poderem dar resposta a uma procura concentrada em apenas alguns dias.

Os próximos cenários servirão para perceber que nível de stock se terá em supermercado e qual a vantagem de ter máquinas a fazer várias referências, avaliar a variação dos tempos de *setup* na taxa de utilização das máquinas e seu impacto no nível de *stock*.

A ordem de mudança de referência é gerada quando o buffer da referência que está a ser produzida está alto e o *buffer* da referência para a qual pode mudar está baixo. Os *buffers* são todos inicializados com 20 caixas.

5.4.1 Dimensionamento dos buffer para uma variação da procura de 50% distribuída

A simulação é de 50 horas, o tempo de trabalho de uma semana em que se trabalha 10 horas por dia.

São duas máquinas em cada exemplo em que cada uma faz uma referência principal mas quando o buffer da referencia da outra máquina está a um nível baixo e o seu está alto esta faz setup e vai fazer a referencia da outra máquina até o nível do buffer da outra máquina esteja razoável.

A soma da procura de cada uma das máquinas é sempre a mesma, sendo a sua média de 100 peças por dia para cada máquina, mas como tem uma variação de 50% em cada dia se a máquina A no primeiro dia faz 150 peças a máquina B faz nesse mesmo dia 50 e vai alterando de dia para dia.

Quando uma máquina faz mais que uma referência um dos factores importantes é o tempo de *setup*, assim foi simulado para uma procura a variar 50% de dia para dia durante a semana e a conclusão a tirar a que para variações até 50% de dia para dia o tempo se *setup* não é relevante porque para um nível de supermercado inicial de 25 caixas, com as mudanças de referência para as condições de o *buffer* da máquina estar acima de 15 e *buffer* da outra máquina estar abaixo de 10 nunca aconteceu e por isso não foi feito nenhum *setup*.

De notar que o buffer da máquina A ficou a 5 e o buffer da máquina B ficou a 12.

5.4.2 Variação da procura em dia sim, dia não.

Neste cenário a procura é feita da seguinte maneira: no primeiro dia apenas a referencia A tem procura, no segundo dia apenas a referencia B tem procura e assim para os resto dos dias da semana.

As máquinas vão fazer um determinado número de setup mediante as condições estabelecidas para tal, essas condições são as seguintes:

- Nível de supermercado inicial 20;
- Se a máquina faz a referência A muda se o seu buffer estiver acima de 15 e o buffer da outra máquina estiver abaixo de 10;
- Número de avarias durante o tempo de simulação 5 de uma hora cada.

Vai ser alterado o tempo de setup para variar de 20% em 20% e ver a taxa de utilização das máquinas e o mínimo atingido em cada *buffer*.

A figura 38 ilustra o supermercado e a maneira como varia o *stock* nos *buffers* para um *setup* de 1.3 hora e assim esta imagem serve como representação dos vários cenários de variação de *setup*.

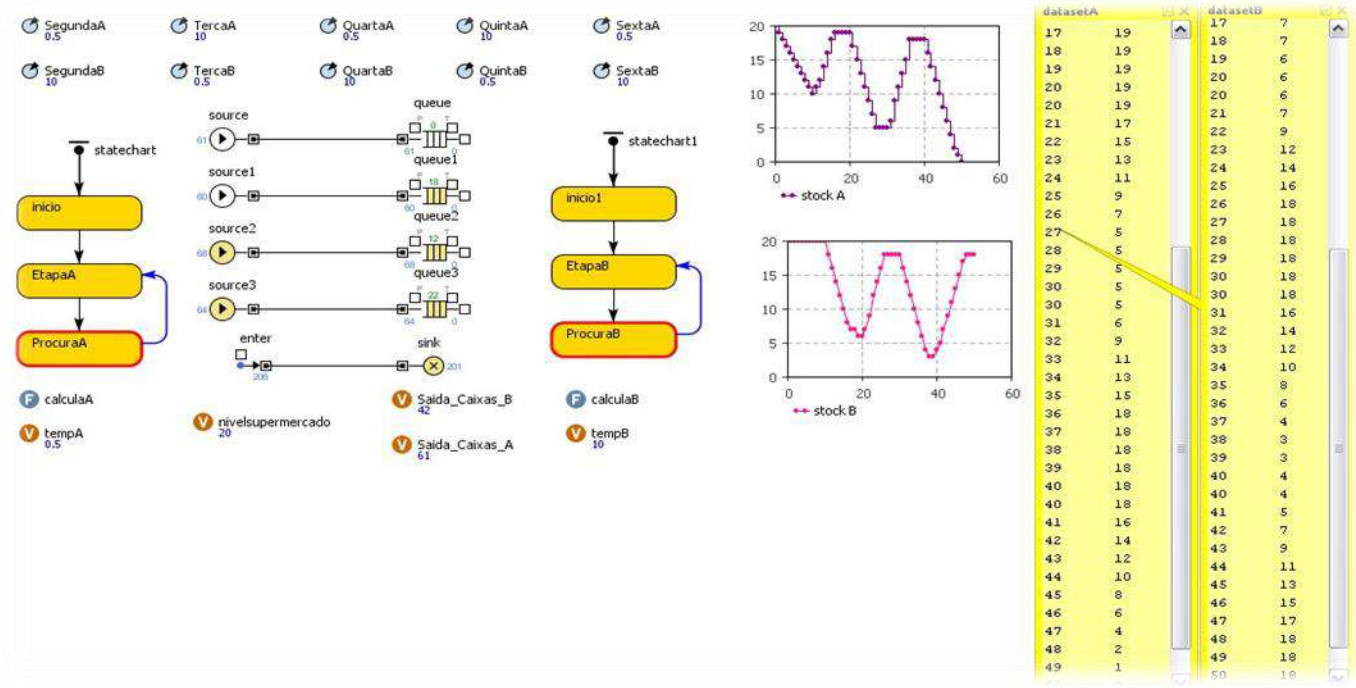

Figura 38 - Buffers para setup de 1.3 hora

A figura 39 ilustra a Máquina A para este cenário com um setup de 1.3 horas.

Aqui é possível observar o número de setups, taxa de utilização da máquina, avarias, limites do *buffer* que servem de condição para poder fazer outras referências, etc.

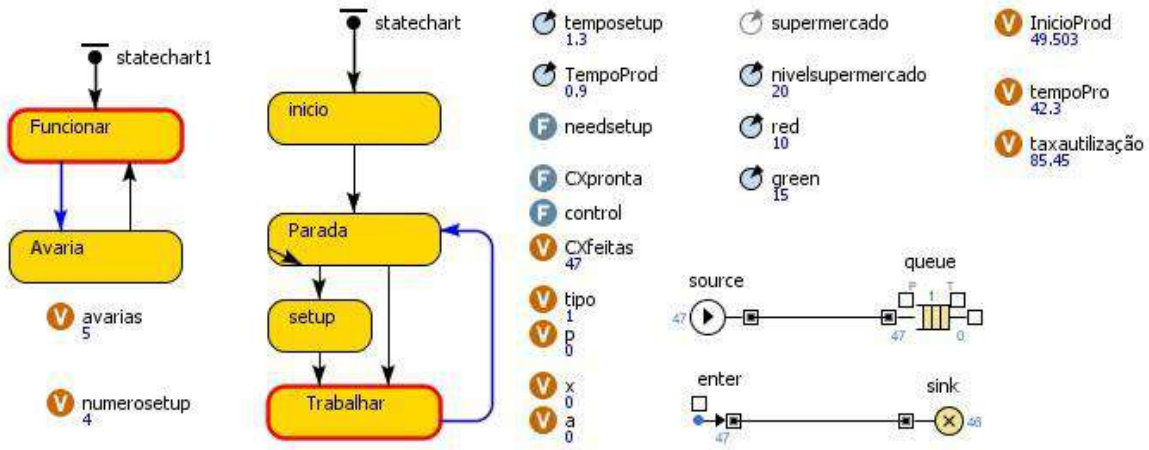

Figura 39 - Maquina A para setup de 1.3 h.

A figura 40 ilustra a máquina B nas condições descritas anteriormente

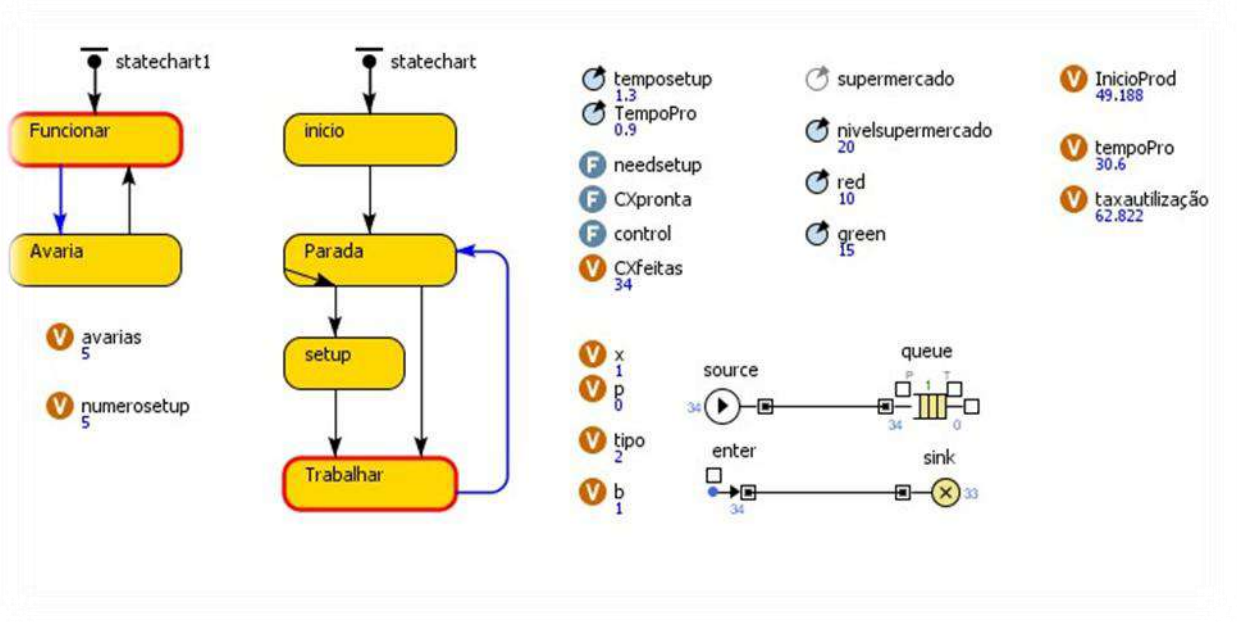

Figura 40 - Máquina B para tempo de setup de 1.3 horas.

Na tabela 5 podemos observar como varia a taxa de utilização para os vários tempos de *setup* bem como os níveis mínimos que atinge cada *buffer* e o número de *setups* que fez cada máquina

Para os tempos de setup de 1.3 horas e 1.5 horas o nível mínimo do *buffer* foi de zero mas esse valor foi atingido apenas no final da simulação e foi logo incrementado ou a simulação chegou mesmo ao fim.

Tabela 5 - Variação dos tempos de setup

Tempos de setup (h)	**Número de setups A**	**Número de setups B**	**Nível mínimo atingido do *buffer* A**	**Nível mínimo atingido do *buffer* B**	**Taxa de utilização da máquina A (%)**	**Taxa de utilização da máquina B (%)**
0.7	4	5	4	6	84,93	69,78
0.9	4	5	1	4	84,31	67,06
1.1	4	5	2	5	85,79	66,79
1.3	4	5	0	3	84,45	62,82
1.5	4	5	0	2	83,29	63,24

5.4.3 Análise de tempos de setup para duas máquinas e três referências.

Este cenário é o mais importante em termos de análise de tempos de setup e utilização das máquinas.

O facto de duas máquinas produzirem três referências provoca vários *setups* ao longo da semana o que torna os tempos de setup muito importantes pois se estes foram demorados a taxa de utilização das máquinas baixa bastante.

Foram feitas várias simulações para vários tempos de setup em que cada máquina faz uma referência principal que tem mais procura mas quando o seu stock está a um nível razoável e o stock de uma outra referência de menor procura está baixo esta máquina vai fazer essa referência até que o nível de *stock* fique razoável ou que o *stock* da sua referência principal fique baixo.

O mesmo acontece com a outra máquina.

A figura 40 representa a evolução dos buffers do supermercado das três referências para um tempo de *setup* de 1 hora.

A regra para produzir a referência com menor procura foi que o seu stock estivesse abaixo de 10 caixas e que os *stocks* das outras referências estivessem acima de 12. Pode ser observado que quando o stock da referência com menor procura chega abaixo de 10 esta começa a ser produzida e o stock das outras duas referências começa a baixar.

Também foi configurada a procura e os tempos de produção de modo a que a produção de cada máquina fosse superior à procura da sua principal referência e assim poder fazer outras referências.

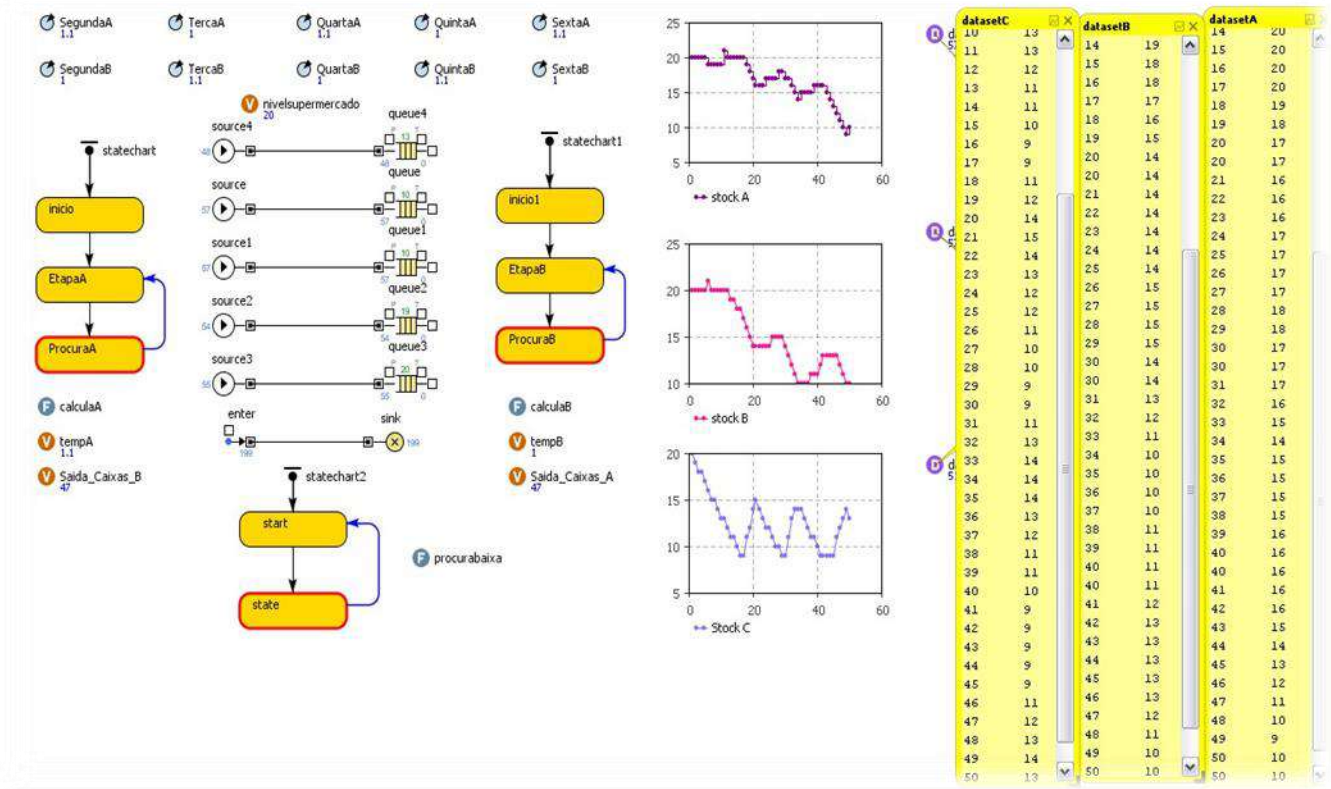

Figura 41 - Evolução do supermercado quando duas máquinas produzem três referências.

A figura 41 ilustra a máquina A onde se pode ver a sua taxa de utilização e o número de setups que têm especial relevo neste cenário pois está a ser estudado o efeito dos tempos de *setup* na taxa de utilização da máquina.

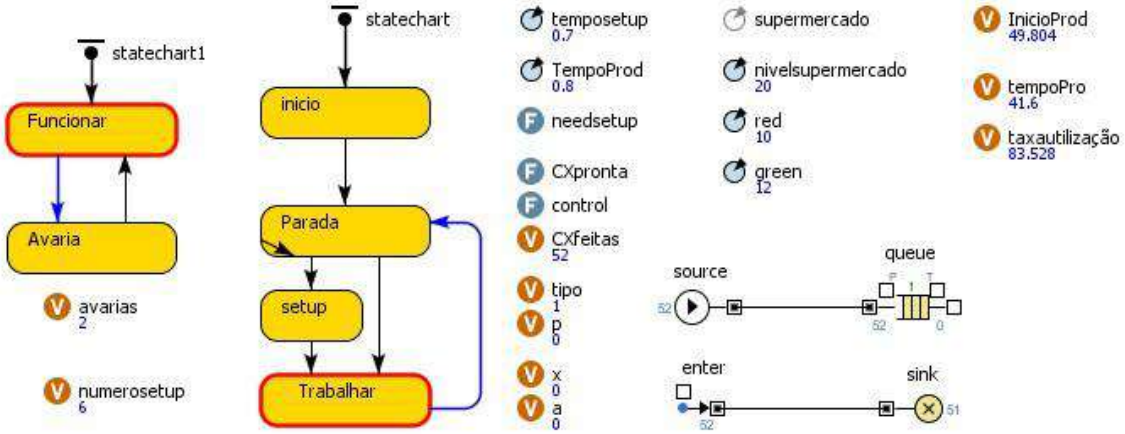

Figura 42 - Máquina A

A figura 42 representa a máquina B que é igual à máquina A mas que teve uma taxa de utilização mais baixa.

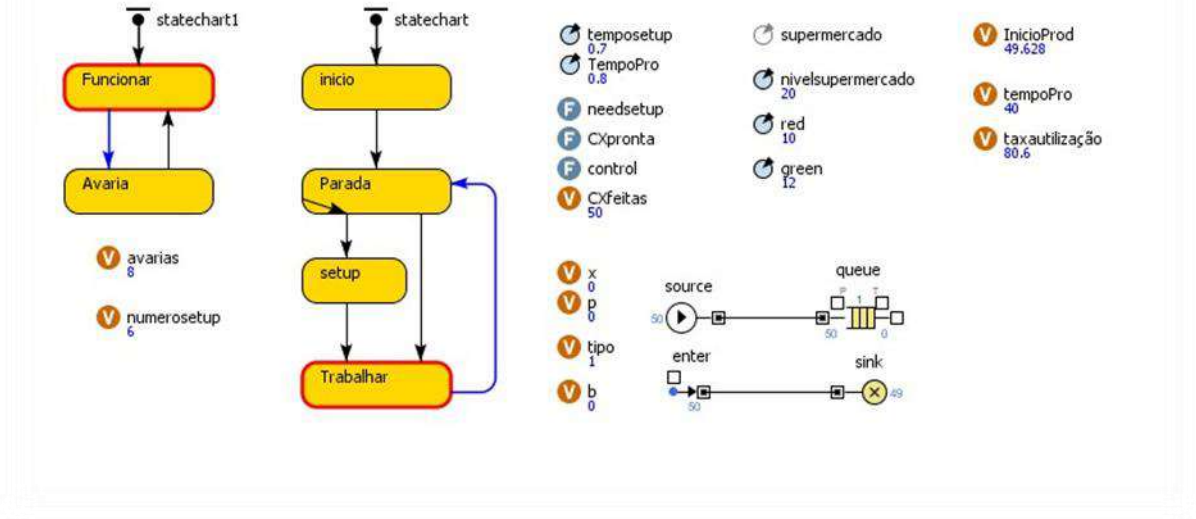

Figura 43 - Máquina B

Na tabela 7 estão os resultados das simulações para vários valores de *setup*, como seria de esperar a taxa de utilização das máquinas baixa quando o tempo de setup aumenta. Por vezes essa variação é menos perceptível quanto a máquina faz o último *setup* já no final da simulação e assim o tempo de *setup* não é diluído na produção seguinte fazendo baixar a taxa de utilização.

Os níveis dos *buffers* foram inicializados com 20 caixas e em nenhum caso se verificou ruptura de stock, ficando estes acima de cinco caixas que foi o valor mínimo atingido no *buffer* da referência C que é a que tem a menor procura.

Tabela 6 - Taxa de utilização e nível dos buffers com a variação de setup

Tempos de setup (h)	**Número de setups A**	**Número de setups B**	**Nível mínimo atingido do *buffer* A**	**Nível mínimo atingido do *buffer* B**	**Nível mínimo atingido do *buffer* B**	**Taxa de utilização da máquina A (%)**	**Taxa de utilização da máquina B (%)**
0.7	6	6	9	10	9	83,53	80,60
0.9	6	6	6	9	8	82,47	77,18
1.1	5	5	8	8	6	80,80	76,95
1.3	5	4	8	9	6	78,87	75,85
1.5	5	4	10	7	5	76,95	75,85

Capitulo 6

6.1 Conclusões e trabalho futuro

Conforme foi afirmado no primeiro capítulo, um dos objectivos deste projecto era conhecer a ferramenta de simulação AnyLogic e avaliá-la na perspectiva da simulação de sistemas de produção, em especial sistemas de produção *lean*.

A ferramenta AnyLogic foi aplicada na simulação de um sistema de produção o que permitiu experienciar a sua capacidade de modelação e avaliar as suas potencialidades na perspectiva da simulação de sistemas de produção. O facto de reunir no mesmo ambiente de desenvolvimento as abordagens de sistemas a eventos discretos, a dinâmica de sistemas e sistemas baseados em agentes, conferiu uma grande flexibilidade na construção dos modelos.

O facto de a ferramenta incluir simulação a eventos discretos, nomeadamente *statecharts*, também se revelou muito útil em vários casos para modelar o comportamento interno das entidades envolvidas no sistema. Graças à *Enterprise Library* do Anylogic é possível modelar problemas, onde existem um fluxo de actividades bem delineado, com grande facilidade, pois os objectos existentes na biblioteca conseguem responder a grande parte das situações que possam ocorrer na realidade

Também é importante realçar o facto de os objectos da *Enterprise Library*, facilitarem criar a animações, com funções próprias para cada objecto, o que também permite separar o modelo interno da visualização.

Da análise dos diferentes cenários criados foi possível chegar a um conjunto de informação que o gestor de produção pode utilizar nas suas decisões e assim melhorar o sistema de produção e avaliar o impacto dos vários cenários no dimensionamento dos buffers do supermercado e da afectação das máquinas a uma ou várias referências.

Assim para procura nivelada em que a procura é próxima da capacidade de produção da máquina a melhor solução é ter máquinas dedicadas a essas referências.

Quando a procura apresenta fortes variações a melhor solução é ter máquinas não dedicadas que podem produzir várias referências para que as máquinas tenham uma melhor taxa de utilização e assim aproveitar os tempos de pouca procura para produzir referências tem tenham *stocks*

reduzidos. Um exemplo disso foi a simulação da procura concentrada no inicio ou final da semana com um stock inicial de 60 caixas, no caso em que as máquinas são dedicadas ocorreu ruptura de *stock* e no caso em que as máquinas não são dedicadas e podem fazer a referência de maior procura não ocorreu ruptura de *stock*.

Uma grande dificuldade na realização do trabalho esteve no facto de não conhecer a realidade da empresa em estudo e assim os cenários de simulação realizados não terem resultados mais concretos.

Nesse sentido deixo algumas ideias para melhorar o sistema de produção, nomeadamente:

- Catalogar todas a referências quanto à variação da procura como, baixa procura, alta procura e procura nivelada com a capacidade de produção;
- Sempre que a procura for nivelada com a produção ou superior deve ter máquinas dedicadas a produzir essas referências;
- As referências com baixa procura devem ser produzidas por máquinas de produção genérica em que estas podem produzir várias referências. Assim quando os stocks estiverem altos fazem outras referências com os stocks reduzidos;
- Catalogar as máquinas quanto à probabilidade de avarias e tempos de *setup*;
- Se possível dedicar as máquinas com tempos de setup altos a produzir referências com procura nivelada ou alta;
- Se possível dedicar as máquinas com maior probabilidade de avaria a produzir referências de baixa procura.
- Elaborar um mapa de manutenção das máquinas de modo a diminuir a probabilidade de avarias;
- Contabilizar as paragens por avaria e tempos de *setup* na taxa de utilização das máquinas e fazer uma análise dos custos dessas paragens;
- Avaliar os custos de paragem por ruptura de *stock* e comparar estes com os custos de paragem das máquinas;
- Avaliar a substituição das máquinas quando estas têm uma taxa de avarias elevada ou os produtos produzidos têm uma taxa de não conformidade alta.

Estas ideias podem ser desenvolvidas em trabalhos futuras com acesso a dados reais e assim utilizar esses dados nos modelos de simulação desenvolvidos ou em novos modelos de simulação.

Referências

C. Dennis Pegden, Robert E. Shannon, and Randall P. Sadowski, “Introduction to Simulation Using Siman”, 1995.

Jeffrey K. Liker, “The Toyota Way: 14 Management Principles from the World's Greatest Manufacturer”, McGraw-Hill, 2004.

Jeffrey K. Liker David Meier, “The Toyota Way Fieldbook”, McGraw-Hill.

R. E. Shannon, “Introduction to the art and science of simulation. Proceedings of the 1998 Winter Simulation Conference”, 1998.

J. S. Carson. Introduction to modeling and simulation. Proceedings of the 2005 Winter Simulation Conference, 2005.

R. Fernandes. Simulador de Sistemas de Produção e de Informação Industriais: Aplicação a sistema de produção lean. Master’s thesis, Faculdade de Engenharia da Universidade do Porto, Julho 2008.

Ferreira, Luís Carlos Ramos Nunes Pinto. Geração Automática de Modelos de Simulação de uma Linha de Produção na Indústria Electrónica. Departamento de Produção e Sistemas, Universidade do Minho. Braga : s.n., 2003.

Hillier, F. S. and Lieberman, G. J. *Introduction to Operations Research*. Boston : McGraw-Hill Higher Education, 2005.

Paquet, Victor and Lin, Li “An Integrated Methodology for Mannufactoring Systems Design Using Manual and Computer Simulation”. Human Factores and Ergonomics in Manufactoring. 2003

Technologies. AnyLogic help. http://www.xjtek.com/anylogic/help/,
Acedido a última vez em Julho de 2009.

XJ Technologies. Discrete event. XJ Technologies. Discrete event, http://www.xjtek.com/anylogic/approaches/discreteevent/
Acedido a última vez em Julho de 2009.